UNIVERSIDADE FEDERAL DE SANTA CATARINA

ULTRAPASSANDO O PODER DE EXPRESSÃO DE SQL COM MINERAÇÃO DE DADOS

KATIANY ZIMMERMANN

FLORIANÓPOLIS
2016/2

UNIVERSIDADE FEDERAL DE SANTA CATARINA

ULTRAPASSANDO O PODER DE EXPRESSÃO DE SQL COM MINERAÇÃO DE DADOS

KATIANY ZIMMERMANN

FLORIANÓPOLIS
2016/2

UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA
CURSO DE SISTEMAS DE INFORMAÇÃO

ULTRAPASSANDO O PODER DE EXPRESSÃO DE SQL COM MINERAÇÃO DE DADOS

KATIANY ZIMMERMANN

Trabalho de conclusão de curso apresentado como parte dos requisitos para obtenção do grau de Bacharel em Sistemas de Informação

FLORIANÓPOLIS
2016/2

KATIANY ZIMMERMANN

ULTRAPASSANDO O PODER DE EXPRESSÃO DE SQL COM MINERAÇÃO DE DADOS

Trabalho de conclusão de curso apresentado como parte dos requisitos para obtenção do grau de Bacharel em Sistemas de Informação.

Orientadora: Carina Friedrich Dorneles

Banca examinadora

Jamil Assreuy Filho

Vania Bogorny

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

Apesar do forte poder de expressão do SQL, que se manifesta através de operadores algébricos e cálculos relacionais, não há suporte para buscas que objetivam descobrir padrões sobre dados de um banco de dados. Com base nisto, este trabalho visa explorar, através de experimentos, até onde é possível obter as informações desejadas sobre um banco de dados fazendo uso exclusivamente de SQL. A partir das limitações encontradas, serão descritas soluções com mineração de dados, de forma a evidenciar motivações para seu uso.

Palavras-chave: SQL. Poder de expressão. Mineração de dados. Padrões em dados.

1 INTRODUÇÃO

Bancos de dados são utilizados em diferentes domínios e aplicações para organizar e armazenar os dados. O uso de banco de dados permite, através de sua organização, controlar a integridade, o acesso e as transações realizadas sobre os dados. Com bancos de dados, a procura de informações através de consultas se torna mais eficiente, possibilitando a programas de computadores buscar dados de forma rápida e simplificada. A representação de tais dados se dá através de modelos. O modelo relacional, por exemplo, é altamente utilizado, uma vez que possui uma definição concreta e madura, oferecendo controle sobre os dados.

Além do uso de programas, a busca de dados em um banco de dados relacional pode ser feita com linguagens de consulta estruturadas (SQL). Sendo um padrão ANSI[1] desde 1986, estruturado pela ISO[2] desde 1987, SQL é considerado um padrão mundialmente. Além disso, não é necessário programar grandes aplicações para usar SQL, que resolve consultas por si de maneira rápida e declarativa para consultar os dados.

Em bancos de dados com características de dados faltantes, ruídos e dados escondidos, SQL não tem poder para extrair análises estatísticas ou descobrir padrões, apesar de sua excelência. Um exemplo real deste cenário é o sistema IPU[3], uma plataforma para consulta a dados cadastrados por membros da UFSC no currículo Lattes (SEGALIN, 2014). Com os dados do IPU, são gerados gráficos indicativos de produção intelectual dos pesquisadores da UFSC (SEGALIN, 2014). Todavia, uma informação que não está explícita no currículo dos pesquisadores é a de departamento, por exemplo. Desta forma, não é possível agrupar os professores por departamento fazendo uso apenas de SQL.

O uso de algoritmos de mineração de dados permite extrair e interpretar padrões dos dados (BOGORNY, 2003). A mineração de dados permite, através de algoritmos automatizados, extrair informações úteis sobre grandes bancos de dados. É preciso, no entanto, conhecer o contexto ao qual se deseja aplicar os algoritmos e entender os dados trabalhados.

---

[1] *American National Standard Institute*. Em português: Instituto Nacional Americano de Padrões.

[2] *International Organization for Standardization*. Em português: Organização Internacional para Padronização.

[3] http://ipu.sistemas.ufsc.br

Considerando as abordagens até então descritas, este trabalho visa realizar experimentos explorando o poder de expressão de SQL, além de aplicar técnicas de mineração de dados em situações não resolvidas por SQL. O sistema IPU é utilizado como ambiente de testes, em que buscas de dados são realizadas através das duas abordagens. Este trabalho tem como contribuição para o estado da arte um estudo referente ao poder de expressão de SQL. Em virtude disso, para buscar respostas em situações não suportadas por SQL, serão aplicados algoritmos de mineração de dados.

1.1 Objetivos

O objetivo geral deste trabalho é estudar o poder de expressão de SQL. Consultas que não são suportadas por SQL serão solucionadas com técnicas de mineração de dados. Para isso, os seguintes objetivos específicos devem ser cumpridos durante seu desenvolvimento:

1. Analisar a fundamentação teórica sobre bancos de dados relacional e mineração de dados.
2. Analisar o estado da arte referente a descoberta de conhecimento com uso de SQL.
3. Buscar soluções com algoritmos de mineração de dados para buscas não suportadas por SQL.

1.2 Organização do trabalho

Este trabalho está organizado em 4 capítulos, que desenvolvem o cumprimento dos objetivos descritos anteriormente. Apresentada a introdução do capítulo 1, o capítulo 2 descreve a fundamentação teórica e trabalhos relacionados, incluindo os conceitos necessários ao entendimento deste contexto e trabalhos existentes que auxiliaram no entendimento do problema em questão. O capítulo 3 demonstra, através de experimentos, o poder de expressão do SQL e exemplos de consultas não suportadas por SQL, de modo a explorar a mineração de dados. Por fim, o capítulo 4 detalha as conclusões obtidas com sugestões de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA E TRABALHOS RELACIONADOS

Para fundamentar o desenvolvimento deste trabalho, a seguir são descritos os conceitos de bancos de dados e mineração de dados, além de uma breve descrição do sistema IPU. Em seguida, é apresentada uma análise de trabalhos relacionados.

2.1 Fundamentação teórica

Nesta seção é apresentada a fundamentação teórica dos principais conceitos que auxiliam o entendimento deste trabalho. Serão apresentados conceitos de bancos de dados e mineração de dados.

2.1.1 Bancos de dados

A busca e a geração de informações relacionadas aos dados de um certo domínio de aplicação se tornam mais eficientes quando tais dados estão organizados. Neste contexto, destaca-se o conceito de bancos de dados, definido como uma coleção de dados relacionados (ELMASRI; NAVATHE, 2010), em que as relações possuem um significado (ELMASRI; NAVATHE, 2010). Elmasri e Navathe (2010) caracterizam um banco de dados como uma representação do mundo real, onde os dados possuem uma lógica coerente e significado próprio, projetado e construído para um objetivo específico.

2.1.1.1 Modelo relacional

Diferentes modelos de dados existem para definir conceitos que descrevem a estrutura de um banco de dados. O modelo de dados relacional é o modelo mais utilizado comercialmente (RAMAKRISHNAN; GEHRKE, 2003, ELMASRI; NAVATHE, 2010). O modelo relacional usa em sua base teórica a teoria de conjuntos e a lógica de predicados (ELMASRI; NAVATHE, 2010).

Dentre os conceitos matemáticos envolvidos em modelos relacionais, destaca-se a relação matemática. Uma coleção de relações em forma de tabelas de valores é utilizada para representar o banco de dados (ELMASRI; NAVATHE, 2010, RAMAKRISHNAN; GEHRKE, 2003), incluindo seus dados e relacionamentos (SUMATHI; ESAKKIRAJAN, 2007). A coleção de relações que representa o banco

de dados no modelo relacional é baseada em uma estrutura. Uma **relação** (ou entidade) se apresenta como uma tabela de valores, com nome único (SILBERSCHATZ; KORTH; SUDARSHAN, 2011). Cada linha (denominada **tupla**) representa uma coleção de valores com dados relacionados entre si, sendo um fato correspondente a uma entidade ou relacionamento do mundo real (ELMASRI; NAVATHE, 2010). Cada coluna é um **atributo** com um tipo definido que caracteriza uma tupla. Se um valor de uma coluna é desconhecido ou não existe, o valor *null*[4] é atribuído ao atributo (SILBERSCHATZ; KORTH; SUDARSHAN, 2011).

Para representar uma tabela de estudantes universitários, a **Tabela 1** mostra uma possibilidade de representação em forma de modelo relacional. Dentre atributos desta tabela que caracterizam um estudante, apresentam-se: cpf, nome, matrícula, universidade e curso que estuda.

Tabela 1: Entidade que representa um estudante.

ESTUDANTE				
CPF	**Matrícula**	**Nome**	**Universidade**	**Curso**
49049304937	00001010	Ana Paula Santos	UFSC	Administração
09493838493	00001011	João Silva	UFSC	Direito
55039554830	00001010	José Paulo Soares	USP	Direito
69388403848	00000998	Maria Paula Machado	USP	Psicologia

Fonte: elaborado pela autora.

Como visto na **Tabela 1**, uma relação pode conter diversas tuplas, possuindo ao menos um atributo que as diferenciam. Os valores dos atributos devem identificar cada tupla de forma única, sendo necessário apenas observar os atributos para estabelecer a distinção entre os registros (SILBERSCHATZ; KORTH; SUDARSHAN, 2011).

Dentre os atributos de uma tabela, podem haver atributos de característica única, definidos como chave primária e chaves únicas. A chave escolhida para identificar um registro de uma relação é a **chave primária** enquanto que as chaves restantes são denominadas **chaves únicas** (ELMASRI; NAVATHE, 2010). Na escolha da chave primária, deve-se levar em consideração um atributo que nunca ou raramente tenha seu valor alterado (SILBERSCHATZ; KORTH; SUDARSHAN, 2011).

Uma relação R1 pode referenciar uma relação R2 através de seus atributos. Neste caso, a chave primária de R2 é um atributo de R1, onde é uma **chave**

[4] Representa um valor não informado ou desconhecido.

estrangeira (SILBERSCHATZ; KORTH; SUDARSHAN, 2011). Na **Tabela 2** por exemplo, a relação descreve características de uma universidade através dos atributos sigla, nome e estado. Na entidade estudante da **Tabela 1**, o atributo 'universidade' pode ser uma chave estrangeira correspondente à chave primária 'sigla' da entidade universidade, na **Tabela 2**.

Tabela 2: Entidade que representa uma universidade.

UNIVERSIDADE		
Sigla	**Nome**	**Estado**
UFSC	Universidade Federal de Santa Catarina	SC
UDESC	Universidade do Estado de Santa Catarina	SC
UFRJ	Universidade Federal do Rio de Janeiro	RJ
UNESP	Universidade Estadual Paulista	SP
USP	Universidade de São Paulo	SP

Fonte: elaborado pela autora.

2.1.1.2 Linguagem SQL

Para obter informações a respeito de dados existentes em um banco de dados, a comunicação pode ocorrer através de consultas ou requisições, realizadas por meio de SQL. Tabelas, linhas e colunas são os termos em SQL que descrevem as relações, tuplas e atributos, respectivamente, originados do modelo formal (ELMASRI; NAVATHE, 2010). A sintaxe de SQL é mais simples de ser compreendida, quando comparada ao modelo formal da álgebra relacional, apresentando expressões para definições de dados, incluindo consultas e atualizações (ELMASRI; NAVATHE, 2010).

A definição dos dados em SQL envolve os comandos e tipos que a linguagem oferece. Os principais comandos e cláusulas de SQL estão descritos na **Tabela 3**, juntamente com algumas funções embutidas, acompanhados por exemplos.

Tabela 3: Comandos, cláusulas e funções de SQL.

Comando	**Descrição**	**Exemplos**
SELECT	Seleciona dados de um banco de dados, em formato de tabela.	**SELECT** nome FROM universidade;
DISTINCT	Elimina linhas com valores repetidos dos atributos especificados na consulta.	SELECT **DISTINCT** sigla FROM universidade.
WHERE	Especifica condições na consulta com base em diferentes operações.	SELECT nome FROM universidade **WHERE** sigla = 'UFSC';
AND	Determina que ambas as condições especificadas na operação sejam verdadeiras.	SELECT cpf FROM estudante WHERE matricula=10201010 **AND** universidade='UFSC';

OR	Determina que ao menos uma das condições especificadas na operação seja verdadeira.	SELECT nome FROM estudante WHERE matricula=10201010 **OR** cpf=92993392999;
ORDER BY	Ordena o resultado de um SELECT pela coluna especificada.	SELECT * FROM curso **ORDER BY** nome;
UNION	União entre resultados de dois ou mais SELECT.	SELECT nome FROM estudante **UNION** SELECT nome FROM professor;
JOIN	Combina linhas de duas ou mais tabelas, com base em uma coluna em comum.	SELECT universidade.nome, estudante.nome FROM universidade **JOIN** estudante ON universidade.sigla = estudante.universidade;
GROUP BY (função)	Agrupa o resultado de funções agregadas no SELECT pela coluna especificada.	SELECT universidade, COUNT(*) FROM estudante **GROUP BY** universidade.
COUNT (função)	Retorna o número de linhas resultantes em um SELECT.	SELECT **COUNT** (*) FROM universidade;
SUM (função)	Retorna a soma dos valores de uma seleção.	SELECT **SUM**(preco) FROM produto;
AVG (função)	Retorna a média dos valores de uma seleção.	SELECT **AVG**(preco) FROM produto;
MAX (função)	Retorna o maior valor da seleção especificada.	SELECT **MAX**(preco) FROM produto;
MIN (função)	Retorna o maior valor da seleção especificada.	SELECT **MIN**(preco) FROM produto;

Fonte: elaborado pela autora.

Para consultar o número de alunos matriculados em cada universidade, por exemplo, é preciso realizar uma seleção (SELECT) que especifica uma operação de contagem (COUNT) sobre o número de alunos agrupados por universidade (com a função GROUP BY), como descreve a consulta abaixo:

```
SELECT COUNT(matrícula) AS soma, universidade
FROM estudante
GROUP BY universidade;
```

Já para consultar dados com valores relacionados em tabelas distintas, pode ser utilizado o operador JOIN. Para incluir, por exemplo, no resultado da consulta acima, a sigla do estado da universidade, é preciso fazer uma operação JOIN da tabela estudante com a tabela universidade. Uma vez que o atributo comum das duas tabelas é a sigla da universidade, este deve ser o atributo equivalente na comparação da junção. Sendo assim, a consulta deve ser como descrita a seguir:

```
SELECT count(matricula) AS soma, universidade, estado
FROM estudante
JOIN universidade
     ON universidade.sigla = estudante.universidade
GROUP BY universidade;
```

Uma consulta que realiza operações sobre os dados e é executada constantemente pode ser armazenada no banco de dados em forma de visão. Em SQL, uma visão é uma tabela que deriva de uma ou mais tabelas e/ou visões (ELMASRI; NAVATHE, 2010). A visão é considerada uma pseudo-tabela, ou tabela virtual, pois não armazena os dados fisicamente, apenas os exibe (SUMATHI; ESAKKIRAJAN, 2007). Com base no exemplo anterior, para criar uma visão no banco de dados que execute a consulta acima, facilitando o acesso a esses dados, a *query* abaixo pode ser executada:

```
CREATE VIEW soma_alunos_universidades AS
     SELECT COUNT(matrícula) AS soma, universidade
     FROM estudante
     GROUP BY universidade;
```

As operações de agregação são requisições comuns em bancos de dados. Tais funções são utilizadas em consultas de estatística que resumem informações, normalmente aplicadas a coleções de valores numéricos (ELMASRI; NAVATHE, 2010), destacadas como função na **Tabela 3**.

Funções como as descritas na **Tabela 3** normalmente já vem implementadas no sistema de banco de dados. Para problemas específicos em um certo contexto, no entanto, é comum que seja necessário implementar uma função, principalmente quando o sistema oferece poucos recursos. A criação de uma função consiste em especificar uma **assinatura**, que inclui o nome e os parâmetros a serem considerados no algoritmo, e uma **implementação**, que é o algoritmo em si (ELMASRI; NAVATHE, 2010). Apesar de SQL não dar suporte, em suas declarações, a ruídos nos dados, por exemplo, algoritmos de similaridade podem auxiliar através de implementações de funções. A medida de Levenshtein, que calcula o menor número de edições necessárias para converter uma *string*[5] A em uma *string* A' (LEVENSHTEIN; 1966), pode ser implementada através de algoritmo em função. Fribble (2006) propõe um algoritmo em SQL que calcula a distância Levenshtein entre duas *strings*, como ilustrado na **Figura 1**.

[5] Sequência de caracteres, utilizadas na área de computação geralmente para representar palavras.

Figura 1: Função que implementa o algoritmo Levenshtein.

```
1  CREATE FUNCTION edit_distance_within(@s nvarchar(4000), @t nvarchar(4000), @d int)
2  RETURNS int
3  AS
4  BEGIN
5    DECLARE @sl int, @tl int, @i int, @j int, @sc nchar, @c int, @c1 int,
6      @cv0 nvarchar(4000), @cv1 nvarchar(4000), @cmin int
7    SELECT @sl = LEN(@s), @tl = LEN(@t), @cv1 = '', @j = 1, @i = 1, @c = 0
8    WHILE @j <= @tl
9      SELECT @cv1 = @cv1 + NCHAR(@j), @j = @j + 1
10   WHILE @i <= @sl
11   BEGIN
12     SELECT  @sc = SUBSTRING(@s, @i, 1),
13             @c1 = @i, @c = @i, @cv0 = '', @j = 1,
14             @cmin = 4000
15     WHILE @j <= @tl
16     BEGIN
17         SET @c = @c + 1
18         SET @c1 = @c1 - CASE WHEN @sc = SUBSTRING(@t, @j, 1) THEN 1 ELSE 0 END
19         IF @c > @c1 SET @c = @c1
20         SET @c1 = UNICODE(SUBSTRING(@cv1, @j, 1)) + 1
21         IF @c > @c1 SET @c = @c1
22         IF @c < @cmin SET @cmin = @c
23         SELECT @cv0 = @cv0 + NCHAR(@c), @j = @j + 1
24     END
25     IF @cmin > @d BREAK
26     SELECT @cv1 = @cv0, @i = @i + 1
27   END
28   RETURN CASE WHEN @cmin <= @d AND @c <= @d THEN @c ELSE -1 END
29 END
```

Fonte: Fribble, SQL Server Forums (2006).

No algoritmo proposto por Fribble (2006), é criada a função com nome `edit_distance_within`, que possui como parâmetros duas *strings* (`@s` e `@t`) de tamanho 4000 e um valor inteiro (@d), retornando um valor inteiro. O algoritmo, descrito entre as palavras BEGIN (quarta linha) e END (última linha), utiliza variáveis auxiliares, declarados como atributos temporários (linhas 5 e 6). Na linha 7, são atribuídos os valores para as variáveis especificadas, como, por exemplo, o tamanho da *string* @s para a variável @s1. Em seguida, são definidas duas repetições condicionais (WHILE). A primeira define o valor de @cv1 de forma interativa. A segunda contém o algoritmo em si, entre as linhas 16 (BEGIN) e 24 (END). Por fim, o algoritmo da função retorna o valor final de @c (similaridade entre os parâmetros) ou -1 (ausência de similaridade).

Com relação aos atributos dos dados, SQL permite definir tipos para cada coluna. Os tipos básicos de dados são: numérico, conjunto de caracteres, booleano e data (ELMASRI; NAVATHE, 2010). Os tipos numéricos incluem números de diferentes tamanhos (INT, SMALLINT) e ponto-flutuante (FLOAT, REAL, DOUBLE PRECISION) (ELMASRI; NAVATHE, 2010). Conjuntos de caracteres podem ter tamanho fixo (CHAR) ou variável (VARCHAR) (ELMASRI; NAVATHE, 2010).

Booleanos incluem os valores TRUE (verdadeiro) e FALSE (falso) (ELMASRI; NAVATHE, 2010). Tipos de data incluem DATA (com 10 posições para ano, mês e dia com separadores: YYYY-MM-DD), TIME (com no mínimo 8 posições para hora, minutos e segundos, podendo incluir separadores: HH:MM:SS) e TIMESTAMP (com as 18 posições anteriores somados a 6 posições para frações de segundos) (ELMASRI; NAVATHE, 2010).

A estrutura da linguagem SQL, incluindo os tipos de dados existentes e as consultas possíveis de se realizar, dão suporte a buscas em banco de dados. É possível, por exemplo realizar comparações entre valores, consultas de valores máximos, mínimos e médios, números de ocorrências, valores distintos, dentre outros tipos de operações disponíveis. Para isso, cada sistema de banco de dados oferece diferentes funções através de SQL, além de permitir que novas funções sejam programadas e criadas. Exemplos de ferramentas disponíveis para gerenciar bancos de dados são: MySQL, SQL Server, Oracle, Sybase e DB2. Uma vez que o sistema IPU utiliza o SQL Server, a mesma ferramenta é utilizada para as consultas realizadas neste trabalho.

2.1.2 Mineração de dados

Com o grande volume de dados nos bancos de dados, é possível gerar novas informações e descrever padrões. O processo de descoberta de conhecimento permite construir um modelo capaz de resolver problemas concretos da vida real (GORUNESCU, 2011). A mineração de dados, que faz parte do processo, auxilia na extração de padrões significativos (ELMASRI; NAVATHE, 2010) por meio de métodos inteligentes capazes de extrair conhecimento de grandes quantidades de dados (SUMATHI; ESAKKIRAJAN, 2007).

Ao aplicar os modelos criados com algoritmos de mineração de dados, são realizadas análises dos padrões obtidos (FAYYAD; PIATETSKI-SHAPIRO; SMYTH, 1996). Na prática, os dois objetivos principais de mineração de dados são a predição e a descrição (FAYYAD; PIATETSKI-SHAPIRO; SMYTH, 1996). A predição usa dados existentes para prever dados desconhecidos ou futuros, enquanto que a descrição encontra padrões que descrevem dados (FAYYAD; PIATETSKI-SHAPIRO; SMYTH, 1996).

Os métodos de mineração de dados são a base da construção do modelo a ser aplicado, caracterizando-se como métodos **supervisionados** e **não-supervisionados**. Em resumo, os métodos supervisionados utilizam um conjunto de dados, chamado treinamento, para deduzir uma função que possa ser aplicada a novos dados de forma a definir seus valores (GORUNESCU, 2011). Já os métodos não-supervisionadas adaptam o modelo de acordo com as observações de forma a agrupar objetos similares (GORUNESCU, 2011). As principais tarefas de mineração de dados, segundo Fayyad, Piatetski-Shapiro e Smyth (1996), são:

- **classificação:** mapeia um dado em uma dentre várias classes pré-definidas;
- **agrupamento:** identifica categorias ou grupos que identificam os dados;
- **associação:** identifica relações entre variáveis;
- **sumarização:** encontra curtas descrições para subconjuntos de dados;
- **detecção de desvios:** identifica dados que diferem do comportamento geral;
- **predição:** mapeia um dado em uma variável de predição com valor real.

Com métodos preditivos e descritivos, os objetivos da mineração são alcançados. Uma vez que os diferentes métodos servem a diferentes propósitos, a escolha do método em uma aplicação deve ser realizada com base no contexto e domínio envolvidos, além do conhecimento que se busca (BOGORNY, 2003).

Dentre as ferramentas disponíveis para o desenvolvimento do processo de mineração de dados, estão: Weka, RapidMiner e R. Weka e R possuem licença GNU[6]. RapidMiner possui licença AGPL[7] com versões gratuita e paga, além de uma versão para programa educacional. Com uma interface amigável, possibilidade de armazenamento em nuvem e intuitiva, a ferramenta RapidMiner foi escolhida para aplicar os processos de descoberta de conhecimento neste trabalho.

Os algoritmos utilizados neste trabalho que implementam as técnicas de mineração acima são descritos a seguir.

2.1.2.1 Classificação

A classificação resulta em conjuntos de modelos (ou funções) de classes ou conceitos. Os modelos gerados são então utilizados para predizer (ou classificar)

[6] *General Public License* - GPL (licença pública geral): licença para *software* livre.

[7] *Affero* GPL: licença para *software* de código aberto.

objetos dos quais a classe seja desconhecida (HAN; KAMBER, 2000). Árvore de decisão, classificação bayesiana e *k-nearest neighbour* (k-NN), ou k-vizinho mais próximo, são alguns dos algoritmos de classificação.

Árvore de decisão (AD): utilizada para predizer a classe ou categoria de um objeto (GORUNESCU, 2011), a árvore de decisão é constituída por nodos (ou nós) e conectados por ramos (CIOS et al, 2007). Cada nodo representa um teste baseado em um certo atributo (GORUNESCU, 2011), exceto pelos nodos inferiores, chamados folhas, que indicam as classes dos objetos, ou decisões (CIOS et al, 2007). Cada ramo representa o resultado de um teste (GORUNESCU, 2011). Dado um conjunto de treinamento, a árvore de decisão realiza particionamentos recursivos de forma a se obter subconjuntos de classe única (BOGORNY, 2003).

Classificação bayesiana: baseada na probabilidade de um evento A ocorrer, dado um evento B ocorrido, segundo a fórmula de Bayes: $P\{B|A\} = P\{A|B\}\times P\{B\} \div P\{A\}$ (GORUNESCU, 2011).

k-NN: classifica um novo dado com base na sua distância com dados classificados (CIOS et al, 2007). A classificação é feita de acordo com os *k* objetos mais próximos (GORUNESCU, 2011).

2.1.2.2 Predição

A predição se difere da classificação por ter como objetivo predizer um valor numérico (HAN; KAMBER, 2000). Um dos algoritmos de predição é a regressão linear.

Regressão Linear: pertencente aos métodos estatísticos, a regressão linear determina a relação linear e o modelo de dados linear para duas ou mais variáveis (CIOS et al, 2007). A regressão múltipla, variável da regressão linear, considera ao menos três variáveis, sendo uma variável dependente e as outras preditoras independentes (GORUNESCU, 2011).

2.2 Trabalhos relacionados

Nesta seção, é apresentado um resumo dos trabalhos de Chaudhari e Khanuja (2015), Wolfram (2006), Ordonez (2006) e Bigolin, Bogorny e Alvares (2003). que auxiliaram no desenvolvimento e entendimento do problema deste trabalho.

2.2.1 Revisão bibliográfica

Com a grande quantidade e diversidade dos dados, a mineração de dados tem ganhado importância em estudos e pesquisas, como nos trabalhos de Chaudhari e Khanuja (2015), Wolfram (2006), Ordonez (2006) e Bigolin, Bogorny e Alvares (2003). Diferentes formas de realizar consultas são aplicadas sobre os respectivos bancos de dados. Para demonstrar as aplicações da mineração de dados, esta seção apresenta um resumo das referências encontradas ressaltando o poder de SQL e o que não é suportado pela linguagem, citados de diferentes formas por Chaudhari e Khanuja (2015), Wolfram (2006), Ordonez (2006) e Bigolin, Bogorny e Alvares (2003).

Na revisão de Chaudhari e Khanuja (2015), SQL é utilizada no preparo de dados para a mineração. Como consequência do esforço no preparo dos dados em um banco relacional para a mineração, o gerenciamento do banco de dados, o desenvolvimento do software envolvido, bem como sua manutenção, se tornam complicados (CHAUDHARI; KHANUJA, 2015). A revisão de Chaudhari e Khanuja (2015) descreve métodos para gerar código SQL que, como resultado, exibem a agregação de colunas em forma de tabela. São destacadas dificuldades relacionadas com funções de agregação de SQL, uma vez que retornam apenas uma coluna por grupos agregados, limitando a construção dos conjuntos a serem gerados para a mineração (CHAUDHARI; KHANUJA, 2015). Uma classe de funções agregadas desenvolvida previamente, denominada agregação horizontal, é vista como solução, retornando um conjunto de números para cada grupo agregado (CHAUDHARI; KHANUJA, 2015). Apesar da solução descrita por Chaudhari e Khanuja (2015), observa-se que os autores utilizam SQL apenas para o preparo dos dados a serem utilizados posteriormente.

No contexto da informetria [8], Wolfram (2006), por sua vez, descreve aplicações de SQL para o processamento da distribuição de frequências informétricas. Apesar das ferramentas oferecidas com uso de SQL para o processamento de dados informétricos, há limitações na eficiência da organização e tabulação dos dados (WOLFRAM, 2006). Existem programas desenvolvidos para tabular características informétricas de conjuntos primários de dados, incluindo

[8] Campo em que técnicas avançadas de recuperação da informação são combinadas com estudos quantitativos de fluxo da informação (WORMELL, 1998).

análises de padrões com mineração de dados (WOLFRAM, 2006). Wolfram (2006) enfatiza a eficiência de ambientes de bancos de dados no processamento dos relacionamentos de distribuição de frequência dos dados gerados por processos de produção de informação. No entanto, as dificuldades de SQL são perceptíveis ao processar tarefas de manipulação de dados complexos (WOLFRAM, 2006).

A fim de demonstrar que é possível realizar tarefas de mineração com SQL, Ordonez (2006) faz uso da linguagem em sua forma pura para implementar algoritmos de agrupamento, integrando-os a bancos de dados. Apesar da implementação sem o uso de extensões com mineração de dados, Ordonez (2006) conclui que a principal dificuldade se dá pela necessidade de, em SQL, ser necessário ler os conjuntos de dados um grande número de vezes, exigindo muito processamento do computador. Ordonez (2006) enfatiza ainda que, pelo tamanho das consultas, é possível que a cláusula CASE apresente problemas.

A fim de automatizar o processo de descoberta de conhecimento, Bigolin, Bogorny e Alvares (2003) apresentam uma linguagem de consulta para bancos espaciais [9] orientados a objetos. Bigolin, Bogorny e Alvares (2003) descrevem extensões de SQL para suportar dados espaciais e não espaciais e para a mineração de dados. As extensões citadas, no entanto, não resolvem todo o processo de mineração de dados, mas permitem selecionar dados geográficos, tratar as informações e extrair conhecimento (BIGOLIN; BOGORNY; ALVARES, 2003). Para automatizar todo o processo, Bigolin, Bogorny e Alvares (2003) definem cláusulas que incluem algoritmos de mineração. Sendo assim, é possível observar no trabalho de Bigolin, Bogorny e Alvares (2003) que a extração de conhecimento em uma base de dados espacial de maneira automática utilizando SQL foi alcançada incluindo algoritmos de mineração.

2.2.2 Análise comparativa

Com base nas referências de Chaudhari e Khanuja (2015), Wolfram (2006), Ordonez (2006) e Bigolin, Bogorny e Alvares (2003), observa-se que o uso de SQL ocorre no preparo dos dados, onde limitações são encontradas e apontadas pelos autores. Uma vez que algoritmos de mineração de dados existem e são amplamente

[9] Bancos de dados com funcionalidades que acompanham objetos de um espaço multidimensional. (ELMASRI; NAVATHE, 2010).

utilizados, o número de pesquisas realizadas que exploram o uso de SQL para a descoberta de conhecimento é limitado. Considerando ainda que SQL é uma linguagem de consulta, novas inferências e descobertas são limitadas, e, quando ocorrem, fazem uso de algoritmos de mineração. A Tabela 4 permite comparar os trabalhos de Chaudhari e Khanuja (2015) [CK], Wolfram (2006) [W], Ordonez (2006) [O] e Bigolin, Borgony e Alvares (2006) [BBA], inclusive com este trabalho [K], através de suas características.

Tabela 4: Comparação dos trabalhos relacionados.

Características	**CK**	**W**	**O**	**BBA**	**K**
Utilizam SQL para obter ou preparar dados	X	X	X	X	X
Enfatizam o poder de expressão de SQL		X	X	X	X
Enfatizam dificuldades encontradas com SQL	X	X	X		X
Desenvolvem funções em SQL	X		X	X	
Citam o uso de ferramentas auxiliares		X			X
Aplicam mineração de dados			X	X	X
Diferenciam uso das duas abordagens					X

Fonte: elaborado pela autora.

3 ANÁLISE EXPERIMENTAL

Esta seção visa descrever a plataforma de testes e construir consultas que explorem o poder de expressão de SQL. Para a construção das consultas, as técnicas de SQL foram exploradas levando à elaboração de enunciados que, hipoteticamente não sejam respondíveis. Os enunciados construídos estão separados em 2 tópicos deste capítulo: consultas respondíveis com SQL e consultas não respondíveis com SQL. Soluções com mineração de dados para as consultas não respondíveis com SQL serão apresentadas no capítulo seguinte.

3.1 Plataforma de testes - IPU

Desenvolvido no trabalho de conclusão de curso de Segalin (2014) para a Pró-Reitoria de Pesquisa da UFSC, o IPU[10] oferece uma visão qualitativa e descritiva das produções da universidade. As informações disponíveis têm como fonte a plataforma Lattes[11].

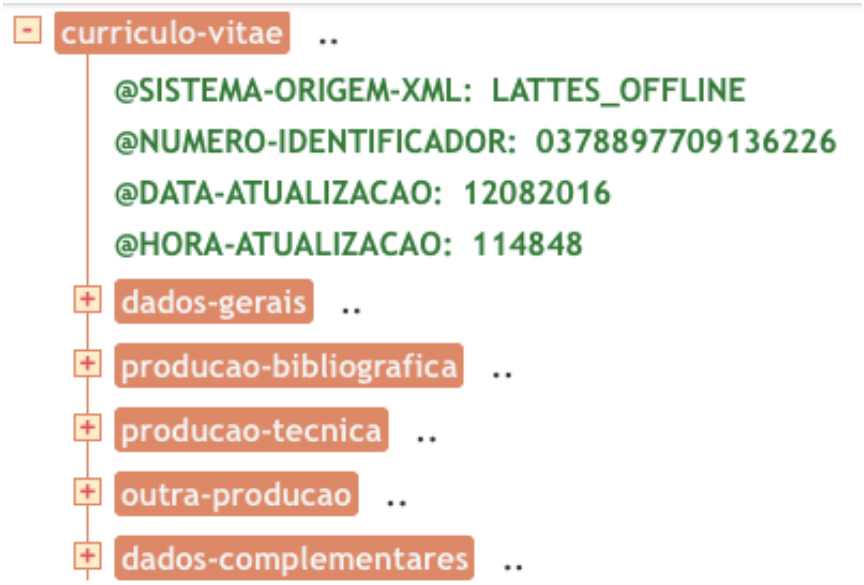
O banco de dados modelado para o IPU foi baseado na estrutura do currículo da plataforma Lattes (SEGALIN, 2014). A partir da obtenção dos currículos Lattes em formato XML, estes são inseridos no banco de dados. O currículo XML tem sua estrutura ilustrada resumidamente na **Figura 2** em forma de árvore. Nesta estrutura resumida, curriculo-vitae, dados-gerais, producao-bibliografica, producao-tecnica, outra-producao e dados-complementares são os nós visíveis, enquanto que os atributos (visíveis na Figura 2 apenas para o nó curriculo-vitae) estão representados através do símbolo @ seguidos de seus respectivos valores.

Com base nessa estrutura, foram criadas, no banco de dados do IPU, uma tabelas para cada nó. A **Figura 3** mostra as tabelas correspondentes aos nós da **Figura 2**. É possível observar que, além do índice e dos atributos, cada tabela (exceto a curriculo_vitae) possui o identificador da tabela curriculo_vitae, indicando um relacionamento.

[10] https://ipu.sistemas.ufsc.br

[11] http://lattes.cnpq.br

Figura 2: Estrutura resumida do XML do currículo Lates.



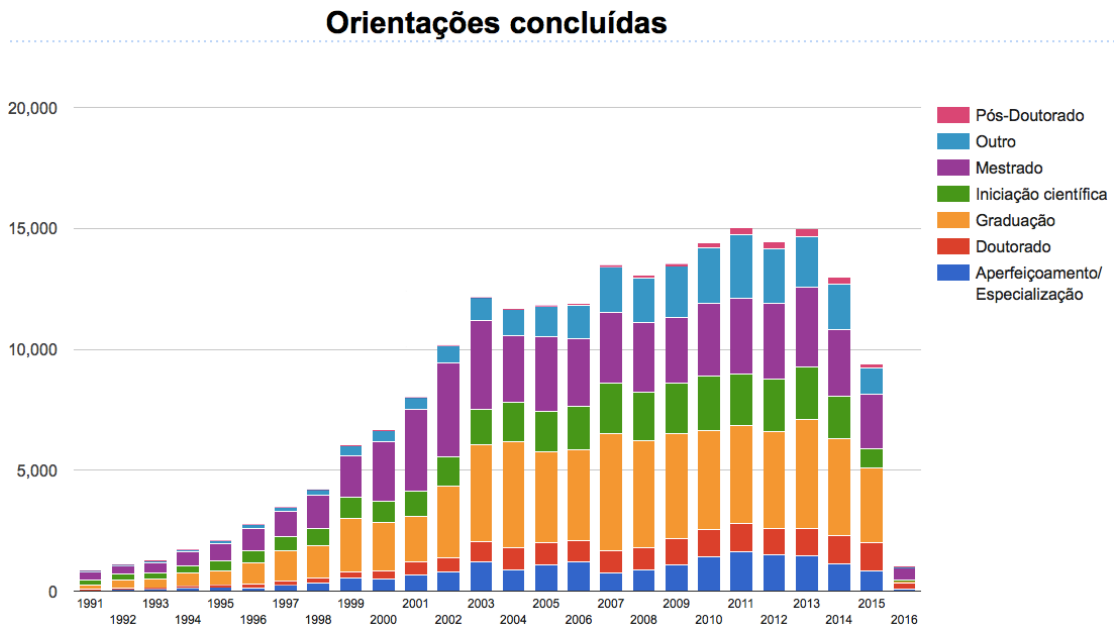
Fonte: elaborado pela autora.

Figura 3: Tabelas do IPU correspondentes à estrutura XML do currículo Lattes.

Fonte: elaborado pela autora.

Com os dados dos currículos distribuídos e organizados no banco de dados modelado para o IPU, consultas são realizadas em SQL (SEGALIN, 2014). Tais consultas extraem os valores de maneira organizada, permitindo que gráficos sejam montados para exibição no IPU. Na seção de indicadores no sistema, por exemplo, os gráficos resumem informações quantitativas extraídas dos currículos dos pesquisadores da UFSC. Os dados da **Figura 4**, por exemplo, são extraídos com uma consulta SQL (Anexo II) desenvolvida por Segalin (2014).

Figura 4: Orientações concluídas por ano.



Fonte: IPU (2016).

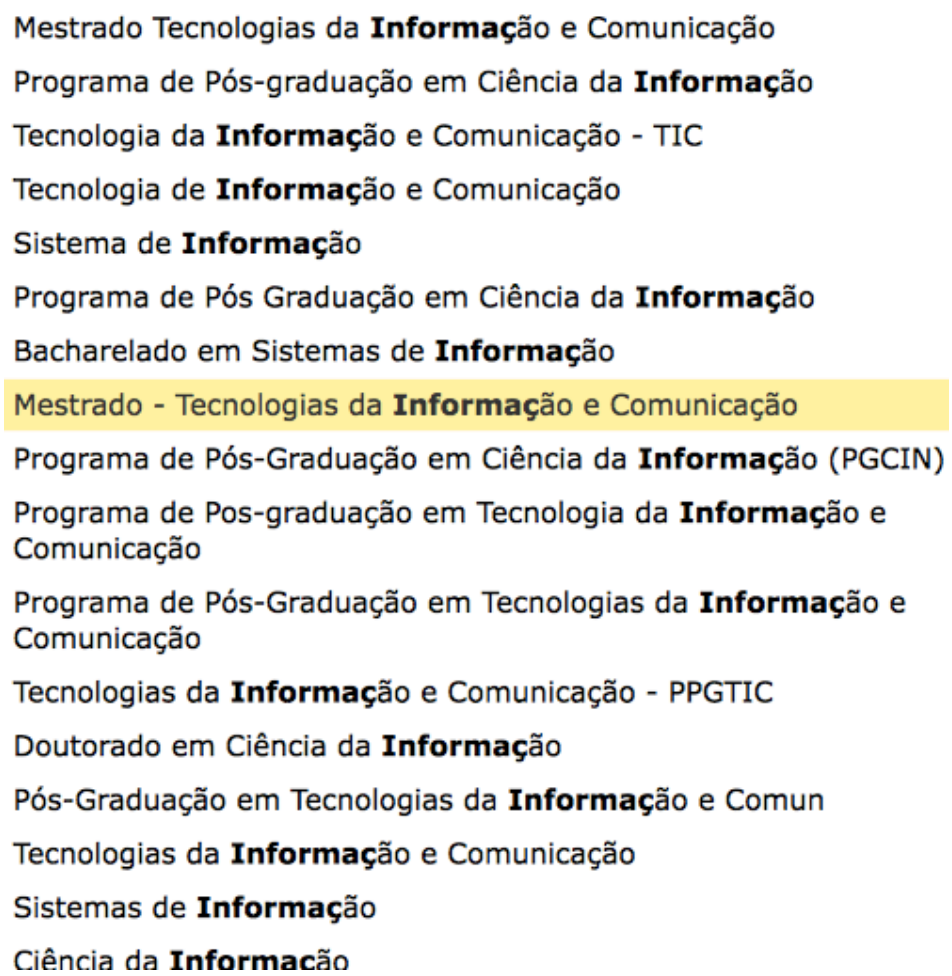
3.1.1 Considerações importantes sobre a plataforma

A seguir, são apresentados alguns desafios de consultas sobre o IPU, com base em seus dados e sua estrutura. Primeiramente, vale ressaltar que os dados do IPU dependem da atualização dos currículos na plataforma Lattes. Caso os pesquisadores da UFSC não atualizem seus currículos constantemente, os números do IPU deixam de refletir a realidade acadêmica da universidade. Dessa forma, pode ser que uma consulta em SQL não seja suficiente para predizer, por exemplo, o número de produções totais com base nos números existentes.

Outra questão refere-se à inserção, de maneira livre, das informações no currículo Lattes, permitindo que os pesquisadores digitem termos quaisquer e informações incompletas em seus currículos. Ao realizar uma busca, no IPU,

referente ao curso de sistemas de informação, por exemplo, a inserção do início da palavra “informação” já mostra uma prévia dos registros existentes, demonstrando que um mesmo curso pode estar definido de diferentes maneiras (**Figura 5**). Com isso, uma determinada consulta em SQL no banco de dados pode não abranger todos os registros existentes. Considerando que as funções e operações oferecidas por SQL não sejam suficientes, algoritmos de mineração de dados baseados em similaridade podem ser explorados como solução.

Figura 5: Busca por termos escritos de diferentes maneiras.



Fonte: IPU (2016).

Do ponto de vista administrativo, pode ser interessante para a universidade analisar as áreas com maior e menor produtividade. Sabendo o número de produções realizadas e agrupando as diferentes áreas, a universidade pode desenvolver programas de incentivo à produção, por exemplo. No entanto, uma consulta em SQL a partir dos dados existentes pode não ser suficiente para definir

os grupos com maior ou menor número de produções. Para isso, pode-se fazer uso de algoritmos de mineração que formem tais grupos e ofereça tais informações.

3.2 Consultas respondíveis com SQL

Nesta seção, são exploradas as técnicas de SQL de modo a demonstrar seu poder de expressão para realizar buscas sobre os dados. Tais técnicas são aplicadas com consultas respondíveis com SQL.

3.2.1 Agregação (ou agrupamento) de valores

Organizar os dados do IPU de forma a extrair, dos últimos cinco anos, apenas os três centros com maior número de produção de patentes na resposta de uma consulta pode ser um desafio em SQL. Neste caso, SQL oferece técnicas de agregação (ou agrupamento) de valores que podem ser exploradas.

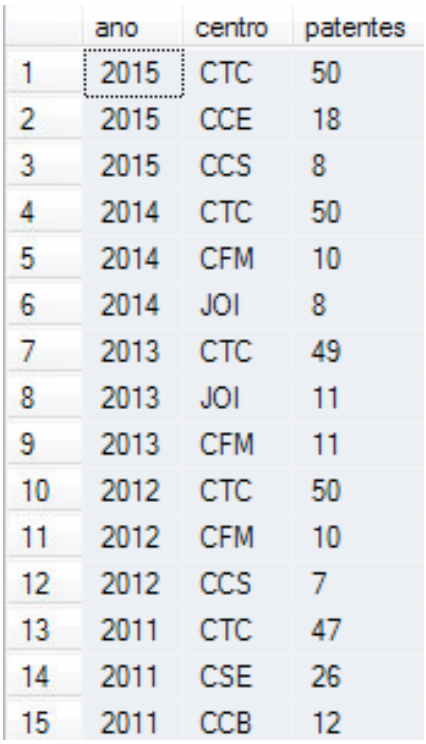
Os dados do IPU necessários neste contexto estão resumidos na visão `patentes`. Dentre outros atributos, a visão inclui o ano em que a patente foi produzida, o título da patente, o tipo de patente e a sigla do centro do pesquisador. Com tais dados, a seguinte *query* SQL retorna os três centros com maior número de patentes nos últimos 5 anos:

```
SELECT ano, centro, patentes FROM (
      SELECT patentes, centro, ano, row_number() OVER
            (PARTITION BY ano ORDER BY patentes DESC) AS rnk
      FROM (
            SELECT count(*) AS patentes, sigla_centro as centro, ano
            FROM lattes.vw_patentes
            GROUP BY sigla_centro
      ) AS pat
      WHERE ano BETWEEEN 2011 AND 2015 AND centro IS NOT NULL) AS res
WHERE rnk <=3
ORDER BY ano DESC, patentes DESC
```

Em resumo, a consulta acima permite, utilizando a visão `patentes`, contar o número de patentes de um ano para cada centro, utilizando as funções `COUNT`, sobre o título da patente e `GROUP BY`, sobre o ano. Para filtrar apenas os anos de 2011 a 2015, utiliza-se a cláusula `WHERE` condicionando que o valor de ano esteja entre 2011 e 2015. Para selecionar apenas os 3 centros que mais produziram patentes, a função `ROW_NUMBER()` é utilizada, associando um número, iniciado em 1, em ordem

crescente, para cada resultado dentro dos diferentes anos, ordenado pela quantidade de produções. Esse número é inserido na cláusula `WHERE` para limitar os resultados até a terceira linha de cada ano. Desta forma, conclui-se que SQL é capaz de extrair os dados do enunciado, conforme resultado da **Figura 6**.

Figura 6: Relatório dos três centros com mais patentes nos últimos cinco anos.



	ano	centro	patentes
1	2015	CTC	50
2	2015	CCE	18
3	2015	CCS	8
4	2014	CTC	50
5	2014	CFM	10
6	2014	JOI	8
7	2013	CTC	49
8	2013	JOI	11
9	2013	CFM	11
10	2012	CTC	50
11	2012	CFM	10
12	2012	CCS	7
13	2011	CTC	47
14	2011	CSE	26
15	2011	CCB	12

Fonte: elaborado pela autora.

3.2.2 Junção e união sobre tabelas distintas que possuem atributo redundante

Outro desafio em SQL pode ocorrer na extração de dados do IPU que não estão explícitos no currículo XML. O número de bolsistas de um professor, por exemplo, não está presente visualmente no currículo. No entanto, existem atributos redundantes em diferentes tabelas que caracterizam projetos com bolsa, vindos do arquivo XML. Sendo assim, técnicas de junção e união de SQL são exploradas de modo que essa informação seja buscada.

Para realizar a busca descrita acima, é possível fazer uma análise sobre os dados do IPU relacionados às orientações e projetos de pesquisa do respectivo

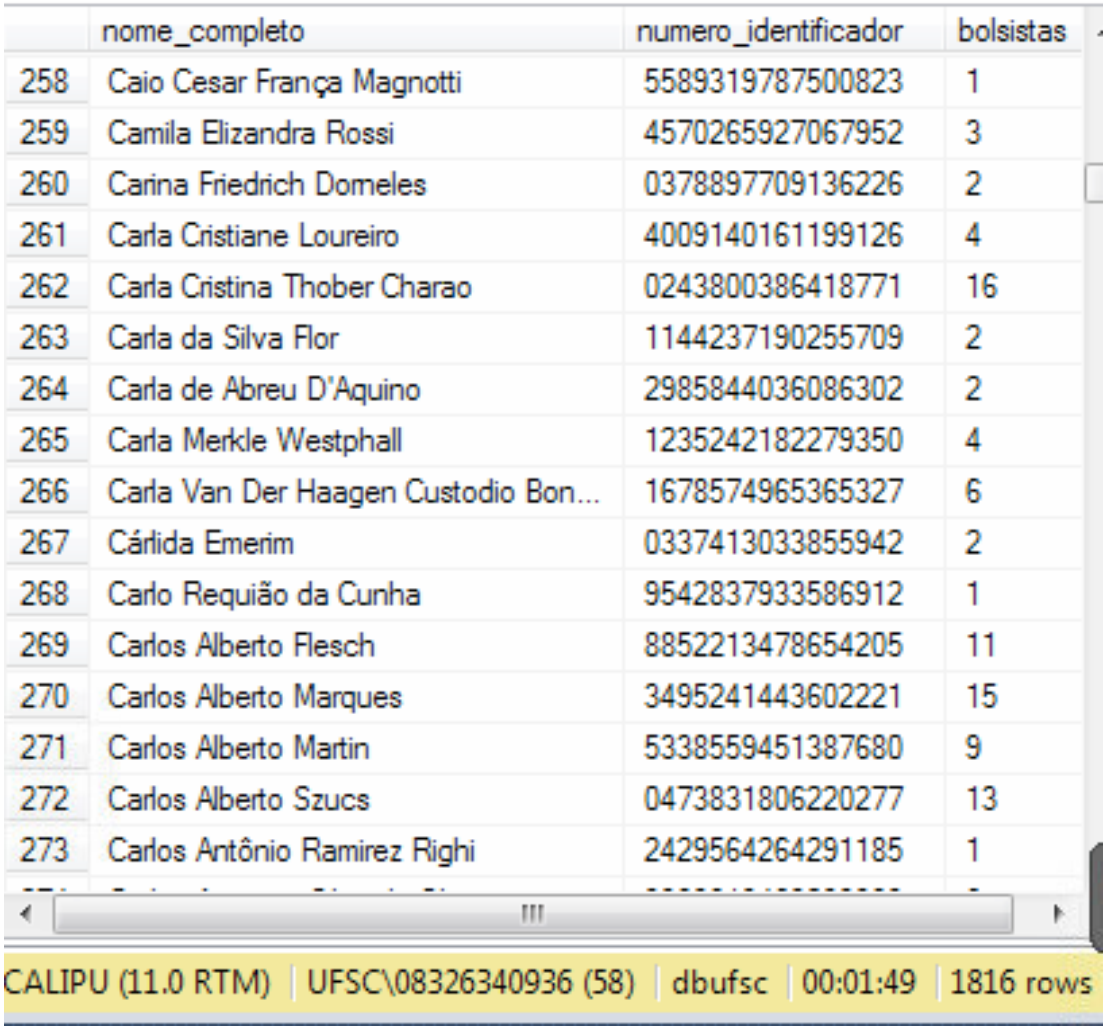
professor. Além da visão vw_projetoPesquisa, com nome e identificador do professor, a tabela financiadores_do_projeto inclui os identificadores de projetos com algum financiamento, caracterizando bolsa projeto de pesquisa com bolsa. Já a tabela detalhamento_de_outras_orientacoes_em_andamento e a visão vw_orientacoesAndamento a contêm os dados de orientações em andamento, incluindo os atributos flag_bolsa (SIM/NÃO) e nome_orientando. A visão vw_orientacoesAndamento possui, além do nome do aluno, o nome do professor e um atributo tipo, que tem o valor Iniciacao Cientifica como um de seus possíveis valores. Com estes dados, a seguinte consulta SQL pode ser realizada:

```
SELECT
s.nome_completo, s.numero_identificador,
SUM(bolsistas) AS bolsistas FROM (
      SELECT
      nome_completo,numero_identificador,COUNT(pp.nome_do_projeto)
      AS bolsistas
            FROM dbufsc.lattes.vw_projetoPesquisa pp
            JOIN lattes.projeto_de_pesquisa prp ON
            prp.nome_do_projeto LIKE pp.nome_do_projeto
            JOIN lattes.financiadores_do_projeto fi ON
            fi.id_projeto_de_pesquisa = prp.id_projeto_de_pesquisa
            WHERE pp.situacao LIKE 'EM_ANDAMENTO'
            GROUP BY nome_completo,numero_identificador
      UNION ALL
      SELECT
      oa.nome_completo, oa.numero_identificador,
      COUNT(oa.nome_do_orientando) AS bolsistas
            FROM dbufsc.lattes.vw_orientacoesAndamento oa
                  JOIN
                  dbufsc.lattes.detalhamento_de_outras_orientacoes_em_
                  andamento doa
                  ON doa.nome_do_orientando = oa.nome_do_orientando
            WHERE doa.flag_bolsa like 'SIM'
            GROUP BY oa.nome_completo, oa.numero_identificador
) s
GROUP BY s.nome_completo,s.numero_identificador
ORDER BY s.nome_completo ASC;
```

A *query* descrita acima realiza uma junção da visão vw_projetoPesquisa com a tabela projeto_de_pesquisa e a tabela financiadores_do_projeto para obter apenas os projetos com bolsa. Com isso, é realizada uma contagem (COUNT) de tais resultados, seguida de um agrupamento (GROUP BY) sobre o nome e número identificador, que caracterizam um professor, considerando o valor EM_ANDAMENTO

do atributo `situacao`. Sobre a visão `vw_orientacoesAndamento`, faz-se uma operação de junção (JOIN) com a tabela de detalhe de orientações onde o valor de `flag_bolsa` é SIM, com base nos valores comuns de `nome_do_orientando`. Com isso, é realizada uma contagem (COUNT) dos nomes dos alunos, agrupando (GROUP BY) pelo nome do professor e número de identificação. O resultado das duas contagens é somado (SUM), agrupando (GROUP BY) novamente pelo nome e pelo número identificador. Para fins de exibição, ao final ainda é feita a ordenação dos resultados pelo nome do professor. A **Figura 7** mostra parte das 1816 linhas resultantes da query SQL.

Figura 7: Relatório do número de bolsistas de um professor.



	nome_completo	numero_identificador	bolsistas
258	Caio Cesar França Magnotti	5589319787500823	1
259	Camila Elizandra Rossi	4570265927067952	3
260	Carina Friedrich Dorneles	0378897709136226	2
261	Carla Cristiane Loureiro	4009140161199126	4
262	Carla Cristina Thober Charao	0243800386418771	16
263	Carla da Silva Flor	1144237190255709	2
264	Carla de Abreu D'Aquino	2985844036086302	2
265	Carla Merkle Westphall	1235242182279350	4
266	Carla Van Der Haagen Custodio Bon...	1678574965365327	6
267	Cárlida Emerim	0337413033855942	2
268	Carlo Requião da Cunha	9542837933586912	1
269	Carlos Alberto Flesch	8852213478654205	11
270	Carlos Alberto Marques	3495241443602221	15
271	Carlos Alberto Martin	5338559451387680	9
272	Carlos Alberto Szucs	0473831806220277	13
273	Carlos Antônio Ramirez Righi	2429564264291185	1

CALIPU (11.0 RTM) | UFSC\08326340936 (58) | dbufsc | 00:01:49 | 1816 rows

Fonte: elaborado pela autora.

3.2.3 Funções de similaridade sobre dados que se referem ao mesmo objeto do mundo real com diferentes representações.

Consultas em SQL que retornam os diferentes valores possíveis de um mesmo objeto pode ser outro desafio. Técnicas de similaridade em SQL são exploradas para obter do IPU o número de orientações concluídas do curso de Sistemas de Informação. Para obter tais resultados, são utilizados os dados da visão `vw_orientacoesConcluidas`, que refletem as informações das orientações concluídas, incluindo o título da orientação e o nome do curso. A consulta a seguir reflete tal busca:

```
SELECT nome_do_curso, count(nome_do_curso) ocorrências
FROM dbufsc.lattes.vw_orientacoesConcluidas
WHERE nome_do_curso like '%sist%informac%'
GROUP BY nome_do_curso
UNION
SELECT n, c
FROM (
      SELECT edit_distance_within('%sistemas de informacao', n,10)
      d, n, c
      FROM (
            SELECT count(nome_do_curso) c, nome_do_curso n
            FROM [dbufsc].[lattes].vw_orientacoesConcluidas
            GROUP BY nome_do_curso
      ) grouped
) similar WHERE d>=0
ORDER BY ocorrencias DESC
```

A função que implementa o algoritmo da distância de Levenshtein (**Figura 1**) é utilizada nesta consulta, considerando os registros de cursos similares à *string* “Sistemas de Informação”. Com tal função, em conjunto com uma simples seleção que considera o valor de nome_do_curso contendo os termos ‘sist’ e ‘informac’, é possível obter um número de 1453 orientações concluídas para o curso especificado.

3.2.4 Combinar atributos nas condições, que, em conjunto, levam à inferência de um resultado esperado

Outra técnica SQL explorada sobre os dados do IPU refere-se à inferência de um resultado através da combinação de diferentes atributos. A busca no IPU por professores da pós-graduação de cada centro pode ser realizada para explorar tal técnica. A informação de que um determinado docente pertence a um programa de pós-graduação não é um dado explícito em uma tabela. Sendo assim, é necessário efetuar a busca sobre diferentes atributos do currículo Lattes para inferir tal informação.

Para realizar tal busca, são encontrados dados de pós-graduação nas visões `vw_orientacoesAndamento`, `vw_conselhoComissaoConsultoria` e `vw_projetoPesquisa` e na tabela `ensino`. Em `ensino`, o atributo `tipo_ensino` possui, dentre seus possíveis valores, o valor `POS-GRADUACAO`. O atributo `tipo` da visão `vw_orientacoesAndamento` pode ter os valores `MESTRADO`, `POS-DOUTORADO` e `DOUTORADO`. A visão `vw_conselhoComissaoConsultoria` possui o atributo `especificacao`, que, dentre seus possíveis valores, possui: `POS-GRADUACAO`, `MESTRADO`, `DOUTORADO` e `POS-DOUTORADO`. Por fim, se um professor tem um projeto de pesquisa e é exibido na visão `vw_projetoPesquisa`, ele é considerado como professor da pós-graduação. O nome do professor não consta na tabela de ensino, sendo necessário buscá-lo na tabela `dados_gerais` através dos relacionamentos de ensino. Já o centro é uma informação armazenada na tabela `professores`. Considerando tais dados, a consulta a seguir extrai, em SQL, informações de professores da pós-graduação:

```
SELECT DISTINCT nome_completo, sigla_centro FROM (
     SELECT DISTINCT nome_completo FROM lattes.ensino en
          JOIN lattes.atividades_de_ensino ae ON
          en.id_atividades_de_ensino = ae.id_atividades_de_ensino
          JOIN lattes.atuacao_profissional ap ON
          ae.id_atuacao_profissional = ap.id_atuacao_profissional
          JOIN lattes.atuacoes_profissionais aps ON
          aps.id_atuacoes_profissionais =
          ap.id_atuacoes_profissionais
          JOIN lattes.dados_gerais dg ON
          dg.id_dados_gerais = aps.id_dados_gerais
     WHERE en.tipo_ensino like 'pos-graduacao'
     UNION
     SELECT DISTINCT nome_completo
```

```
      FROM lattes.vw_orientacoesAndamento
      WHERE tipo IN ('mestrado', 'pos-doutorado', 'doutorado')
      UNION
      SELECT DISTINCT nome_completo FROM lattes.vw_projetoPesquisa
      UNION
      SELECT DISTINCT nome_completo
      FROM lattes.vw_conselhoComissaoConsultoria
      WHERE especificacao LIKE '%pos%graduacao%'
            OR especificacao LIKE '%mestrado%'
            OR especificacao like '%doutorado%'
) x
LEFT JOIN lattes.professores p ON x.nome_completo = p.nome
WHERE sigla_centro IS NOT NULL
ORDER BY sigla_centro, nome_completo;
```

Nesta consulta, é realizada uma operação de união (UNION) entre os resultados das seleções dos professores na tabela `ensino` e as visões `vw_orientacoesAndamento`, `vw_projetoPesquisa` e `vw_conselhoComissaoConsultoria` que possuem, dentre os atributos especificados anteriormente, dados de pós-graduação. Com isso, obtém-se o nome de todos os professores com atividades de pós-graduação. Por fim, é realizado um JOIN com a tabela de professores, com base no nome do professor, para selecionar o valor de centro. A consulta resulta em 2596 resultados.

3.3 Consultas não respondíveis com SQL

Buscas que dependem de técnicas de predição e classificação de dados desconhecidos são apresentados nesta seção. Para cada técnica, são apresentados enunciados que ultrapassam o poder de expressão de SQL, explorando os ruídos dos dados presentes no IPU.

3.3.1 Classificação de dados desconhecidos

EDC1: Classificar o centro, quando ausente, ao qual pertence uma orientação que foi concluída.

Origem do problema: O IPU possui uma tabela de professores, gerada a partir de dados extraídos de sistemas da SeTIC[12] para controle e cadastro dos professores da UFSC. Tal tabela possui a informação do centro de cada professor e é

[12] Superintendência de Governança Eletrônica e Tecnologia da Informação e Comunicação.

relacionada com outras tabelas do IPU, com base no nome do professor, que é o único dado comum entre os dois sistemas. Uma vez que existem, no IPU, pesquisadores não vinculados à UFSC que orientam alunos da UFSC, o valor de centro é ausente, sendo registrado no banco de dados como NULL.

Limitação encontrada: Predições não são resolvidas com SQL, pois, tecnicamente, é uma linguagem que efetua tarefas de extração de dados existentes no banco de dados, incluindo operações sobre os dados, resumindo as informações em formato de relatório. Sendo assim, SQL não consegue reproduzir dados que não estejam registrados no banco de dados.

EDC2: Gerar um relatório de todos os professores por departamento.

Origem do problema: Nos currículos Lattes, não há um campo de preenchimento específico para departamento. Uma análise dos dados permitiu concluir que alguns professores preenchem tal informação em diferentes partes do currículo, como no resumo, no endereço profissional e em vínculos. Com uma consulta em SQL, é possível obter a informação de departamento de 2304 currículos, de um total de 4738 currículos existente na base de dados. Ou seja, apenas 48% dos professores registram o seu departamento no currículo, em locais diferentes.

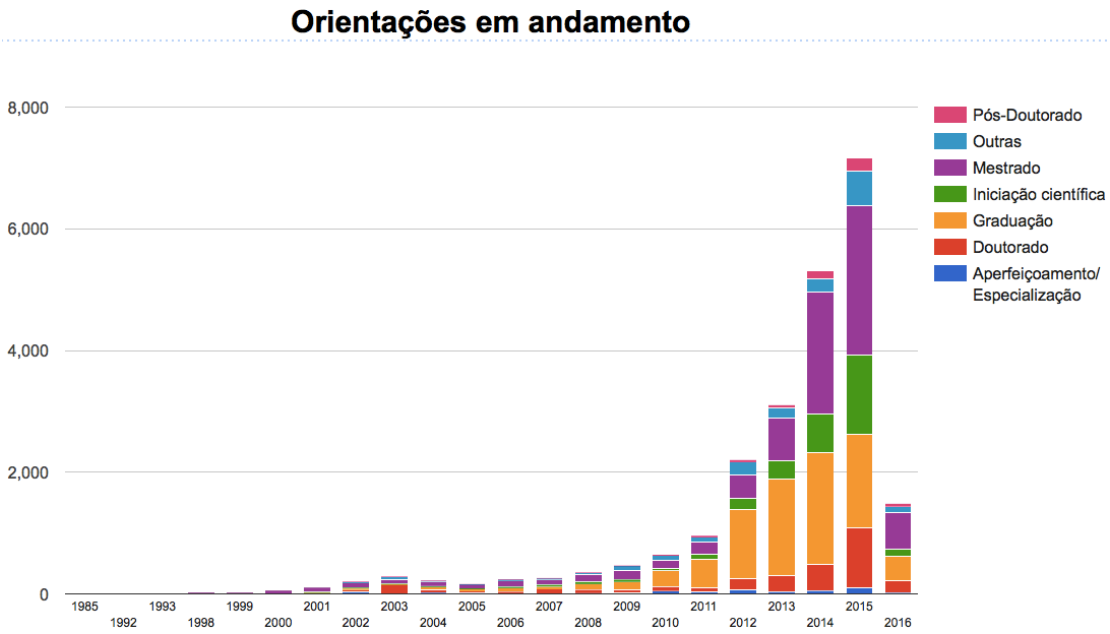
Limitação encontrada: Apesar de haver informações de departamento no banco de dados do IPU, não é possível, com SQL, extrair o departamento de 52% dos professores.

3.3.2 Predição de valores futuros ou desconhecidos

EDC3: Gerar um relatório mais preciso do número de orientações concluídas em 2015.

Origem do problema: No gráfico da **Figura 4**, é possível perceber que o número de orientações concluídas em 2015 não está de acordo com o número de orientações concluídas dos anos anteriores. A falta de atualização constante das informações no Lattes faz com que tais valores não sejam coerentes. Observando a **Figura 8**, é possível observar um número considerável de orientações em andamento em anos anteriores a 2016, provavelmente devido à falta de atualização dos currículos. Com base na **Figura 4** e na **Figura 8**, fica evidente que o número de orientações concluídas em 2015 não está coerente com os números dos anos anteriores.

Figura 8: Orientações em andamento por ano.



Fonte: IPU (2016).

Limitação encontrada: Os relatórios gerados em SQL consomem os dados registrados no banco de dados. Se os dados não são atualizados, os resultados refletidos estarão também desatualizados. Neste caso, é preciso que a atualização seja feita pelo usuário, uma vez que, com SQL não é possível realizar operações de predição.

EDC4: Predizer o número de orientações concluídas ao final de 2016.

Origem do problema: Uma vez que 2016 é o ano atual, orientações que estão por ser concluídas constam ainda como “em andamento” no IPU. No entanto, a falta de atualizações dos currículos faz com que o número de orientações, tanto em andamento como concluídas, exibido no IPU, não sejam suficientes para saber o número de orientações concluídas ao final de 2016. Para ter uma estimativa do número de orientações que serão concluídas ao final do ano, é preciso fazer predições. No IPU, as orientações em andamento

Limitação encontrada: Assim como descrito na limitação do problema anterior (EDC3), não é possível, tecnicamente, com SQL, predizer valores de dados e informações que não existem no banco de dados.

3.4 Soluções com mineração de dados

A seguir, são propostas soluções para os enunciados do tópico anterior, utilizando algoritmos de mineração de dados com a ferramenta RapidMiner. Serão aplicadas as técnicas de classificação e predição, descritas na seção 2.1.2 deste trabalho.

3.4.1 EDC1: Classificar orientações concluídas por centros

Para este problema, três algoritmos diferentes de classificação foram aplicados: árvore de decisão (AD), classificação bayesiana e k-NN.
Dados extraídos do IPU: `vw_orientacoesConcluidas`.

Solução 1: AD.
Pré-processamento: exclusão de atributos irrelevantes; remoção de registros duplicados; filtro de registros onde centro não é NULL.
Mineração: aplicação do algoritmo de árvore de decisão para gerar o modelo, com 3 segundos de execução.
Modelo gerado: classifica um registro com base nos atributos ano (valor numérico dos anos) e tipo (com valores categorizados em Doutorado, Graduacao e Mestrado). Uma vez que estes atributos não são suficientes para determinar o valor de centro, o modelo AD gerado foi descartado.

Solução 2: classificador bayesiano.
Pré-processamento: seleção de atributos relevantes (nome, instituição, curso, centro, tipo); remoção de registros duplicados; filtro de registros onde centro não é NULL.
Mineração: aplicação do operador de classificação bayesiana para gerar o modelo, com 5 segundos de execução.
Modelo gerado: foram criadas 5 distribuições para as classes (centros) definidas, com base nos atributos relevantes para o algoritmo. O modelo gerado resultou nas probabilidades de cada centro descritas na **Tabela 5**.

Tabela 5: Modelo bayesiano de probabilidade de cada centro.

Centro	P	Centro	P
CTC	0.181	CBS	0.012
CCB	0.076	CSE	0.078
CFM	0.058	CAC	0.007
CCE	0.1	CA	0.013
CCA	0.056	ARA	0.017
CED	0.071	CASGO	0.004
CCJ	0.035	PRDHS	0.002
CCS	0.139	JOI	0.017
CFH	0.11	BLN	0
CDS	0.025		

Fonte: elaborado pela autora.

Pós-processamento: exclusão de atributos irrelevantes gerados pelo modelo; remoção de registros duplicados; seleção do centro com maior número de atribuições ao professor.

Observações do resultado: uma vez que o modelo pode, com base nas probabilidades dos atributos, predizer valores de centros diferentes para um mesmo professor, distribuídos uniformemente, optou-se por manter todos os valores de centro resultantes. Por exemplo, na **Figura 9**, observa-se que, para a professora Janaina Goncalves Guimaraes, foi predito o valor de três possíveis centros. Tais predições possuem o mesmo peso, com base nas orientações registradas no banco de dados. Nestes casos, o modelo poderia ainda, com base nas probabilidades da **Tabela 3**, extrair apenas o centro com maior probabilidade.

Figura 9: Amostra do resultado da predição do centro com classificador bayesiano.

Row No.	nome_completo	pred_centro
353	Jaceny Maria Reynaud	CASGO
354	Jairo Nunes de Freitas	BLN
355	Jakerson Ricardo Gevinski	CSE
356	Jamil Assreuy Filho	CCB
357	Janaina Goncalves Guimaraes	ARA
358	Janaina Goncalves Guimaraes	CCJ
359	Janaina Goncalves Guimaraes	CTC
360	Jane da Silva	ARA
361	Janeide Freitas de Mello	CCS
362	Janete Kaminski	BLN
363	Janete Kaminski	CSE

Fonte: elaborado pela autora.

Solução 3: k-NN.

Pré-processamento: seleção de atributos relevantes (nome, instituição, curso, centro, tipo); remoção de registros duplicados; filtro de registros onde centro não é NULL.

Mineração: aplicação do operador k-NN para gerar o modelo, com 18 segundos de execução para k=2.

Modelo gerado: a ferramenta informa apenas que o modelo resulta em 20.343 exemplos com 4 dimensões em cada classe existente.

Pós-processamento: exclusão de atributos irrelevantes gerados pelo modelo; remoção de registros duplicados; seleção do centro com maior número de atribuições ao professor.

Observações do resultado: apesar de o tempo de execução ser 3,6 vezes o tempo do classificador bayesiano, observa-se, na **Figura 10**, que as predições são mais precisas com k-NN. Neste caso, com k=2, o algoritmo é capaz de filtrar até dois vizinhos de cada registro que apresentam maior similaridade.

Figura 10: Amostra do resultado da predição de centro com k-NN.

Row No.	nome_completo	pred_centro
352	Jaceny Maria Reynaud	CSE
353	Jairo Nunes de Freitas	CCA
354	Jakerson Ricardo Gevinski	CTC
355	Jamil Assreuy Filho	CCB
356	Janaina Goncalves Guimaraes	CTC
357	Jane da Silva	CCS
358	Janeide Freitas de Mello	CCS
359	Janete Kaminski	CSE
360	Jean Carlo Rossa Hauck	CTC

Fonte: elaborado pela autora.

3.4.2 EDC2: Classificar professores em departamentos

Para este problema, dois métodos diferentes foram aplicados: AD e classificação bayesiana.

Dados extraídos do IPU: `endereco`, `endereco_profissional`, `dados_gerais`, `vw_orientacoesAndamento`, `vw_orientacoesConcluidas`.

Solução 1: AD.

Pré-processamento: seleção de atributos de `endereco_profissional` com valor de departamento; filtro de atributos relevantes de `vw_orientacoesConcluidas` e `vw_orientacosAndamento`; seleção de registros da UFSC com valor de departamento para conjunto de treinamento.

Mineração: aplicação do algoritmo AD para gerar modelo, com 1 segundo e 4 segundos de execução.

Modelo gerado: neste caso, o algoritmo conseguiu gerar a árvore apenas com base no valor de centro, devido ao formato dos atributos relevantes. Sendo assim, o modelo AD gerado foi descartado.

Solução 2:

Pré-processamento: seleção de atributos de `endereco_profissional` com valor de departamento; filtro de atributos relevantes de `vw_orientacoesConcluidas` e `vw_orientacosAndamento`; junção dos resultados obtidos; remoção de dados duplicados; limpeza *strings* de departamento; seleção de registros da UFSC com valores válidos de curso e departamento para conjunto de treinamento.

Mineração: aplicação do classificador bayesiano para gerar o modelo, com 51 segundos de execução.

Modelo gerado: o modelo gerado (em anexo) resultou em pelo menos 139 classes com probabilidades > 0, sendo 73 classes com probabilidade > 0.002 e 17 mais relevantes com probabilidade > 0.01.

Pós-processamento: exclusão de atributos irrelevantes gerados pelo modelo.

Observações do resultado: antes de aplicar o algoritmo, o maior número de professores com valor de departamento registrado no banco se concentrava no departamento de química. Com a aplicação do algoritmo mineração de dados, por sua vez, os grupos de departamentos ficaram mais distribuídos.

3.4.3 EDC3: Predizer número de orientações concluídas de 2015

Para este problema, foi aplicado o método de regressão linear.

Dados extraídos do IPU: `vw_orientacoesConcluidas` e `vw_orientacoesConcluidas.`

Solução: regressão linear.

Pré-processamento: seleção de atributos relevantes de orientações (ano, titulo do trabalho); agrupamentos para obter a contagem de orientações por ano; construção de uma tabela única com número de orientações em andamento e número de orientações concluídas por ano.

Mineração: partição dos dados em conjunto de treinamento (90% dos dados, com exemplares lineares para realizar o aprendizado ao decorrer dos anos) e conjunto de teste; aplicação do algoritmo de regressão linear para gerar modelo, com 2 segundos de execução.

Pós-processamento: junção dos resultados calculados com os valores existentes no IPU para comparação.

Observações do resultado: uma vez que o conjunto de treinamento foi separado com 90% dos registros, o cálculo de orientações concluídas foi realizado apenas para o ano de 2015, resultando em 12582 orientações, como mostra a **Figura 11**. O cálculo poderia ser realizado para outros anos, diminuindo o conjunto de treinamento e ampliando o conjunto de teste. Poderia ser considerado, ainda, o número de professores admitidos e desligados da UFSC por ano, o que auxiliaria no aprendizado do algoritmo para efetuar as predições. No entanto, essa informação não está disponível no IPU.

Figura 11: Predição das orientações concluídas em 2015.

ano	andamento	concluidas	prediction...
2010	640	14405	?
2011	960	15050	?
2012	2207	14462	?
2013	3109	14989	?
2014	5319	12965	?
2015	7173	9413	12582.847

Fonte: elaborado pela autoral.

3.4.4 EDC4: Predizer número de orientações concluídas em 2016

Para este problema, foi aplicado o método de regressão linear.

Dados extraídos do IPU: `vw_orientacoesConcluidas` e `vw_orientacoesConcluidas.`

Solução: regressão linear.

Pré-processamento: seleção de atributos relevantes de orientações (ano, titulo do trabalho); agrupamentos para obter a contagem de orientações por ano; construção de uma tabela única com número de orientações em andamento e número de orientações concluídas por ano.

Mineração: partição dos dados em conjunto de treinamento (95% dos dados, com exemplares lineares para realizar o aprendizado ao decorrer dos anos) e conjunto de

teste; aplicação do algoritmo de regressão linear para gerar modelo, com 5 segundos de execução.

Pós-processamento: junção dos resultados calculados com os valores existentes no IPU para comparação.

Observações do resultado: como mostra a **Figura 12**, o algoritmo conseguiu predizer um valor de 9.834 orientações concluídas ao final de 2016. No entanto, observou-se, no enunciado anterior, que o número de orientações concluídas de 2015 pode ser maior do que o presente no banco de dados, devido a falta de atualizações dos currículos Lattes. Desta forma, vale adaptar as configurações da etapa de mineração para considerar um conjunto de treinamento menor, aumentando o conjunto de teste para aplicar as previsões desde 2015. Neste caso, com um conjunto de treinamento correspondente a 90% dos registros, a **Figura 13** resulta em um número consideravelmente maior de 17.792 orientações concluídas em 2016. Assim como no enunciado anterior, o número de professores admitidos e desligados da UFSC poderiam auxiliar o aprendizado do algoritmo.

Figura 12: Predição das orientações concluídas em 2016.

ano	andamento	concluidas	prediction(c...
2011	960	15050	?
2012	2207	14462	?
2013	3109	14989	?
2014	5319	12965	?
2015	7173	9413	?
2016	1491	1011	9834.619

Fonte: elaborado pela autora.

Figura 13: Predição das orientações concluídas em 2015 e 2016.

ano	andamento	concluidas	prediction...
2010	640	14405	?
2011	960	15050	?
2012	2207	14462	?
2013	3109	14989	?
2014	5319	12965	?
2015	7173	9413	12582.847
2016	1491	1011	17972.673

Fonte: elaborado pela autora.

3.5 Análise e discussão entre abordagens

Considerando os enunciados propostos e as soluções obtidas, vale ressaltar as vantagens e desvantagens das abordagens utilizadas. Apesar da simplicidade e agilidade providos através do poder de expressão de SQL, a diversidade e complexidade dos recursos de mineração de dados deve ser ressaltada.

Dos enunciados propostos resolvidos com SQL, a maior parte do esforço se dá pela construção das consultas. Com visões construídas que resumem e operam sobre os valores existentes, tais consultas se tornam mais simples. No entanto, a necessidade de realizar operações com lógicas mais complexas é facilitada com a construção de funções. Combinando funções e operadores SQL, é possível gerar relatórios sobre os dados existentes de forma eficiente. Tais características são evidentes nos enunciados resolvidos por SQL, que combinam o uso de operadores JOIN, UNION e funções em SQL.

Na ausência de valores em um banco de dados, por sua vez, SQL se limita com o preenchimento de valores padrões ou simplesmente nulos. A descoberta de conhecimento requer o uso de mineração de dados. A aplicação dos algoritmos de mineração provêm a descoberta de padrões para suprir os valores ausentes com informações relevantes. Por outro lado, é necessário realizar um esforço maior na etapa de pré-processamento e ter conhecimento sobre os algoritmos existentes. Com o desenvolvimento das soluções para os enunciados criados, ficou evidente

que a dedicação maior foi aplicada ao preparo e organização dos dados para, então, serem submetidos à mineração.

Com base nisto, o uso de SQL é suficiente, em diferentes ocasiões. Quando os registros necessários para extrair as informações desejadas estão presentes no banco de dados, SQL é a abordagem recomendada. Agrupamentos e agregações de valores também podem ser realizados através de SQL. Funções de similaridade também podem ser implementadas através de SQL. Por outro lado, há casos em que não é possível realizar agrupamentos ou buscas por similaridades, mas que podem ser resolvidos através de técnicas de mineração de dados.

Apesar das possibilidades com mineração de dados, diferentes técnicas e algoritmos resultam em diferentes conclusões. Em EDC1 e EDC2, por exemplo, foi preciso aplicar mais de um algoritmo para conseguir gerar um modelo apropriado. Técnicas não aplicadas poderiam resultar, ainda, em outras conclusões. No entanto, algumas técnicas requerem alto poder de processamento e memória em um computador.

4 CONCLUSÕES E TRABALHOS FUTUROS

A transformação dos dados em informação e conhecimento deve ser prioridade em um ambiente de banco de dados. Grandes volumes de dados requerem ambientes bem estruturados, o que inclui a organização dos dados e o conhecimento das ferramentas disponíveis. Um ambiente de desenvolvimento pode iniciar basicamente com um banco de dados e um sistema de acesso à informação. Com o aumento da quantidade de dados, pode ser conveniente ampliar tal ambiente, aderindo novas ferramentas e explorando diferentes conceitos, como a mineração de dados.

Em um sistema de gerenciamento de banco de dados, SQL conta alto poder de expressão para extrair dados. Incontáveis consultas podem ser realizadas sobre um banco de dados utilizando declarações com operadores e funções de SQL. No entanto, apesar de uma estrutura bem definida de um banco de dados, dados inconsistentes podem começar a surgir de modo que SQL deixe de ser uma solução.

O uso de outras abordagens pode ultrapassar o poder de expressão de SQL. A mineração de dados, através de diferentes algoritmos, é capaz de aprender os padrões implícitos nos dados e realizar descoberta de conhecimento. Estudos relacionados mostram que o uso de SQL ocorre até o momento do preparo dos dados, para, a partir de então, a aplicação de novos conceitos. Os dados são obtidos através de consultas SQL para então serem utilizados com mineração de dados. Sendo assim, os casos que ultrapassam o poder de SQL permitem explorar outras abordagens, em diferentes etapas.

Através dos experimentos realizados explorando SQL e mineração de dados, este trabalho corrobora com o estado da arte. Apesar das dificuldades encontradas em SQL, as possibilidades de consultas e operações sobre os dados são inúmeras. Com grandes quantidades de dados, SQL e mineração de dados são duas abordagens diferentes, capazes de extrair dados de diferentes maneiras.

Como trabalhos futuros, novas pesquisas, experimentos e aplicações podem ser realizadas. No IPU, por exemplo, podem ser incluídas ferramentas de mineração, de modo a gerar novos relatórios com análises qualitativas e quantitativas, auxiliando na tomada de decisão. Outros dados podem ser vinculados ao IPU, aperfeiçoando o aprendizado dos algoritmos. Diferentes técnicas que

exploram poder de processamento e memória computacionais podem ser aplicadas para obter melhores resultados.

REFERÊNCIAS

BIGOLIN, Nara Martini; BOGORNY, Vania; ALVARES, Luis Otávio. Uma Linguagem de Consulta para Mineração de Dados em Banco de Dados Geográficos Orientado a Objetos. In: **Conferencia Latinoamericana de Informatica**. 2003. p. 23-35.

BOGORNY, Vania. **Algoritmos e ferramentas de descoberta de conhecimento em bancos de dados geográficos**. 2003. Dissertação - PPGC, UFRGS, Porto Alegre, 2003.

CHAUDHARI, Archana A.; KHANUJA, Harmeet Kaur. Database Transformation to Build Data-set for Data Mining Analysis-A Review. In: **Computing Communication Control and Automation (ICCUBEA), 2015 International Conference on**. IEEE, 2015. p. 386-389.

CIOS, Krzysztof J.; PEDRYCZ, Witold; SWINIARSKI, Roman W. Data Mining and Knowledge Discovery. In: **Data Mining Methods for Knowledge Discovery**. Springer US, 2007.

ELMASRI, Ramez. **Fundamentals of database systems**. Pearson Education India, 2010.

FAYYAD, Usama; PIATETSKY-SHAPIRO, Gregory; SMYTH, Padhraic. From data mining to knowledge discovery in databases. **AI magazine**, v. 17, n. 3, p. 37, 1996.

FRIBBLE, Arnold. SQL Server Forums – Levenshtein Edit Distance Algorithm, 2005. Disponível em: <http://www.sqlteam.com/forums/topic.asp?TOPIC_ID=51540>. Acesso em: 15 de ago. 2016.

GORUNESCU, Florin. **Data Mining: Concepts, models and techniques**. Springer Science & Business Media, 2011.

HAN, Jiawei; KAMBER, Micheline. **Data mining concepts and techniques**. Simon Fraser University, 2000.

LEVENSHTEIN, Vladimir I. Binary codes capable of correcting deletions, insertions, and reversals. Soviet Physics Doklady, v. 10, p. 707, 1966.

ORDONEZ, Carlos. Integrating K-means clustering with a relational DBMS using SQL. **IEEE transactions on Knowledge and Data engineering**, v. 18, n. 2, p. 188-201, 2006.

RAMAKRISHNAN, Raghu; GEHRKE, Johannes. **Database management systems**. McGraw-Hill, 2003.

SEGALIN, Vinicius S. **Ipu: sistema web de acesso a dados de Currículos Lattes a partir do mapeamento XML-Relacional dos currículos de professores e**

pesquisadores da UFSC. 2014. Dissertação (Graduação em Sistemas de Informação) - UFSC, Florianópolis, 2014.

SUMATHI, Sai; ESAKKIRAJAN, S. **Fundamentals of relational database management systems**. Springer, 2007.

SILBERSCHATZ, Abraham et al. **Database system concepts**. New York: McGraw-Hill, 2011.

WOLFRAM, Dietmar. Applications of SQL for informetric frequency distribution processing. **Scientometrics**, v. 67, n. 2, p. 301-313, 2006.

WORMELL, Irene. Informetria: explorando bases de dados como instrumentos de análise. **Ciência da Informação**, v. 27, n. 2, p. 210-216, 1998.

ANEXOS

Anexo I

Classes geradas pelo classificador bayesiano para o modelo a ser aplicado na predição de departamento.

Classe	Probabilidade
departamento de enfermagem	0.024
departamento de engenharia mecânica	0.022
departamento de química	0.016
departamento de psicologia	0.016
departamento de língua e literatura vernáculas	0.014
departamento de ciências da administração	0.013
departamento de geociências	0.013
departamento de engenharia elétrica	0.013
departamento de saúde pública	0.013
departamento de informática e estatística	0.013
departamento de engenharia civil	0.012
departamento de língua e literatura estrangeiras	0.012
departamento de metodologia de ensino	0.012
departamento de educação física	0.011
departamento de bioquímica	0.01
departamento de nutrição	0.01
departamento de matemática	0.01
departamento de ciências contábeis	0.009
departamento de física	0.009
departamento de estomatologia	0.009
departamento de expressão gráfica	0.008
departamento de automação e sistemas	0.008
departamento de ciências econômicas	0.008
departamento de antropologia	0.008
departamento de engenharia química	0.007
departamento de fitotecnia	0.007
departamento de análises clínicas	0.007
departamento de clínica cirúrgica	0.007
departamento de história	0.007
departamento de filosofia	0.007
departamento de engenharia de produção	0.007
departamento de arquitetura	0.006
departamento de botânica	0.006
departamento de estudos especializados em educação	0.006
departamento de ciência da informação	0.006
departamento de ciência e tecnologia dos alimentos	0.006
departamento de ciências farmacêuticas	0.005
departamento de engenharia sanitária e ambiental	0.005
departamento de aquicultura	0.005
departamento de odontologia	0.005
departamento de ciências fisiológicas	0.004
departamento de clínica médica	0.004
departamento de serviço social	0.004

colégio de aplicação	0.004
departamento de engenharia sanitária	0.004
departamento de engenharia rural	0.004
departamento de engenharia do conhecimento	0.004
departamento de zootecnia	0.004
departamento de microbiologia, imunologia e parasitologia	0.004
departamento de sociologia e ciência política	0.004
departamento de farmacologia	0.004
departamento de engenharia de produção e sistemas	0.004
departamento de biologia celular, embriologia e genética	0.003
departamento de direito	0.003
departamento de patologia	0.003
departamento de ecologia e zoologia	0.003
departamento de microbiologia e parasitologia	0.003
departamento de ciências sociais	0.003
departamento de informática e de estatística	0.003
departamento de zootecnia e desenvolvimento rural	0.003
departamento de arquitetura e urbanismo	0.003
departamento de engenharia química e engenharia de alimentos	0.003
departamento de jornalismo	0.003
departamento de ciências biológ. e da saúde e de ciências soc. aplicadas	0.003
departamento de biologia celular embriologia e genética	0.002
núcleo de desenvolvimento infantil	0.002
departamento de pediatria	0.002
departamento de comunicação e jornalismo	0.002
departamento de biologia	0.002
departamento de bioquimica	0.002
departamento de ciência e tecnologia de alimentos	0.002
departamento de economia e relações internacionais	0.002
departamento de aquicultura, centro de ciências agrárias	0.002
departamento de administração	0.001
departamento de fisica	0.001
departamento de morfologia	0.001
departamento de tocoginecologia	0.001
departamento de patologia médica	0.001
departamento de engenharia quimica e engenharia de alimentos	0.001
departamento de lingüística e filologia	0.001
departamento de direito público e ciência política	0.001
departamento de automação	0.001
departamento de ciencias farmaceuticas	0.001
departamento de ciências morfológicas	0.001
departamento de recursos humanos	0.001
departamento de línguas e literaturas estrangeiras	0.001
departamento de engenharia mecanica	0.001
departamento de ecologia e zoologia - ecz	0.001
departamento de expressão gráfica - egr	0.001
departamento de métodos e técnicas	0.001
departamento de medicina	0.001
departamento de metodologia do ensino e educação comparada	0.001

departamento de biblioteconomia e documentação	0.001
departamento de artes e libras	0.001
departamento de ciências de administração	0.001
departamento de engenharia química e de alimentos	0.001
departamento de economia	0.001
departamento de odontologia preventiva	0.001
departamento de obras hidráulicas	0.001
departamento de lingüística	0.001
departamento de biologia celular embriologia e genática	0.001
departamento de automacao e sistemas	0.001
departamento de letras estrangeiras	0.001
departamento de aqüicultura	0.001
departamento de ciencia e tecnologia de alimentos	0.001
departamento de ciências da administração - cad	0.001
departamento de metodologia - men	0.001
departamento de línguas e literaturas estrangeiras - cce	0.001
centro de ciências da educação - colégio de aplicação	0.001
departamento de saúde`pública	0.001
departamento de solos	0.001
departamento de ecologia e zoologia - ccb	0.001
departamento de engenharia mecânica - laboratório de vibrações e acústica	0.001
departamento de ciências fisiologicas do centro de ciências biológicas	0.001
departamento de geociências - cfh	0.001
departamento de sociologia e ciências políticas	0.001
departamento de ecologia e zoologia (ecz)	0.001
departamento de ginecologia e obstetrícia	0.001
departamento de sociologia e ciencia politica	0.001
departamento da ciencia da informação	0.001
departamento de ecologia e zoologia/ccb	0.001
departamento de estudos especializados em educação - eed	0.001
departamento de geociencias ufsc	0.001
departamento de letras e literaturas vernáculas llv	0.001
curso de fonoaudiologia	0.001
departamento de ciências sociais aplicadas	0.001
departamento de botânica - centro de ciências biológicas	0.001
departamento de ciências da administração/cad	0.001
departamento de informática	0.001
departamento de física / cfm	0.001
departamento de administraçao	0.001
departamento de saúde pública, centro de ciências de saúde	0.001
departamento de ensino superior	0.001
departamento de enegnharia química e engenharia de alimentos	0.001
departamento de ciência da informação - cin	0.001
departamento de engenharia e gestão do conhecimento	0.001
departamento interdisciplinar do câmpus litoral norte	0.001
departamento de engenharia e gestão do conhecimento- ufsc	0.001
departamento de expressão gráfica ? egr	0.001

Anexo II

Consulta SQL que extrai informações das orientações em andamento no IPU (Segalin, 2014).

```
SELECT ROW_NUMBER() OVER (ORDER BY titulo, ano, nome_do_orientado,
nome_da_instituicao, nome_do_curso, tipo, nome_completo, numero_identificador,
sigla_centro) AS row, titulo, ano, nome_do_orientado, nome_da_instituicao,
nome_do_curso, tipo, nome_completo, numero_identificador, sigla_centro FROM (
SELECT dboc.titulo, dboc.ano, detoc.nome_do_orientado, detoc.nome_da_instituicao,
detoc.nome_do_curso, 'Mestrado' AS tipo, dg.nome_completo,
cv.numero_identificador, p.sigla_centro
FROM lattes.orientacoes_concluidas_para_mestrado oc
JOIN lattes.dados_basicos_de_orientacoes_concluidas_para_mestrado dboc ON
dboc.id_dados_basicos_de_orientacoes_concluidas_para_mestrado =
oc.id_dados_basicos_de_orientacoes_concluidas_para_mestradoconcluido
JOIN lattes.detalhamento_de_orientacoes_concluidas_para_mestrado detoc ON
detoc.id_detalhamento_de_orientacoes_concluidas_para_mestrado =
oc.id_detalhamento_de_orientacoes_concluidas_para_mestradoconcluido
JOIN lattes.orientacoes_concluidas ocs ON ocs.id_orientacoes_concluidas =
oc.id_orientacoes_concluidas
JOIN lattes.outra_producao op ON op.id_outra_producao = ocs.id_outra_producao
JOIN lattes.curriculo_vitae cv ON cv.id_curriculo_vitae = op.id_curriculo_vitae
JOIN lattes.dados_gerais dg ON dg.id_curriculo_vitae = cv.id_curriculo_vitae
LEFT JOIN lattes.professores p ON LOWER(dg.nome_completo) collate
sql_latin1_general_cp1251_cs_as = LOWER(p.nome) collate
sql_latin1_general_cp1251_cs_as
UNION ALL
SELECT dboc.titulo, dboc.ano, detoc.nome_do_orientado, detoc.nome_da_instituicao,
detoc.nome_do_curso, 'Doutorado' AS tipo, dg.nome_completo,
cv.numero_identificador, p.sigla_centro
FROM lattes.orientacoes_concluidas_para_doutorado oc
JOIN lattes.dados_basicos_de_orientacoes_concluidas_para_doutorado dboc ON
dboc.id_dados_basicos_de_orientacoes_concluidas_para_doutorado =
oc.id_dados_basicos_de_orientacoes_concluidas_para_doutoradoconcluido
JOIN lattes.detalhamento_de_orientacoes_concluidas_para_doutorado detoc ON
detoc.id_detalhamento_de_orientacoes_concluidas_para_doutorado =
oc.id_detalhamento_de_orientacoes_concluidas_para_doutoradoconcluido
JOIN lattes.orientacoes_concluidas ocs ON ocs.id_orientacoes_concluidas =
oc.id_orientacoes_concluidas
JOIN lattes.outra_producao op ON op.id_outra_producao = ocs.id_outra_producao
JOIN lattes.curriculo_vitae cv ON cv.id_curriculo_vitae = op.id_curriculo_vitae
JOIN lattes.dados_gerais dg ON dg.id_curriculo_vitae = cv.id_curriculo_vitae
LEFT JOIN lattes.professores p ON LOWER(dg.nome_completo) collate
sql_latin1_general_cp1251_cs_as = LOWER(p.nome) collate
sql_latin1_general_cp1251_cs_as
UNION ALL
SELECT dboc.titulo, dboc.ano, detoc.nome_do_orientado, detoc.nome_da_instituicao,
detoc.nome_do_curso, 'Pos-Doutorado' AS tipo, dg.nome_completo,
cv.numero_identificador, p.sigla_centro
FROM lattes.orientacoes_concluidas_para_pos_doutorado oc
JOIN lattes.dados_basicos_de_orientacoes_concluidas_para_pos_doutorado dboc ON
dboc.id_dados_basicos_de_orientacoes_concluidas_para_pos_doutorado =
oc.id_dados_basicos_de_orientacoes_concluidas_para_pos_doutorado
JOIN lattes.detalhamento_de_orientacoes_concluidas_para_pos_doutorado detoc ON
detoc.id_detalhamento_de_orientacoes_concluidas_para_pos_doutorado =
oc.id_detalhamento_de_orientacoes_concluidas_para_pos_doutorado
```

```
JOIN lattes.orientacoes_concluidas ocs ON ocs.id_orientacoes_concluidas =
oc.id_orientacoes_concluidas
JOIN lattes.outra_producao op ON op.id_outra_producao = ocs.id_outra_producao
JOIN lattes.curriculo_vitae cv ON cv.id_curriculo_vitae = op.id_curriculo_vitae
JOIN lattes.dados_gerais dg ON dg.id_curriculo_vitae = cv.id_curriculo_vitae
LEFT JOIN lattes.professores p ON LOWER(dg.nome_completo) collate
sql_latin1_general_cp1251_cs_as = LOWER(p.nome) collate
sql_latin1_general_cp1251_cs_as
UNION ALL
SELECT dboc.titulo, dboc.ano, detoc.nome_do_orientado, detoc.nome_da_instituicao,
detoc.nome_do_curso, 'Aperfeicoamento e especializacao' AS tipo, dg.nome_completo,
cv.numero_identificador, p.sigla_centro
FROM lattes.outras_orientacoes_concluidas oc
JOIN lattes.dados_basicos_de_outras_orientacoes_concluidas dboc ON
dboc.id_dados_basicos_de_outras_orientacoes_concluidas =
oc.id_dados_basicos_de_outras_orientacoes_concluidas
JOIN lattes.detalhamento_de_outras_orientacoes_concluidas detoc ON
detoc.id_detalhamento_de_outras_orientacoes_concluidas =
oc.id_detalhamento_de_outras_orientacoes_concluidas
JOIN lattes.orientacoes_concluidas ocs ON ocs.id_orientacoes_concluidas =
oc.id_orientacoes_concluidas
JOIN lattes.outra_producao op ON op.id_outra_producao = ocs.id_outra_producao
JOIN lattes.curriculo_vitae cv ON cv.id_curriculo_vitae = op.id_curriculo_vitae
JOIN lattes.dados_gerais dg ON dg.id_curriculo_vitae = cv.id_curriculo_vitae
LEFT JOIN lattes.professores p ON LOWER(dg.nome_completo) collate
sql_latin1_general_cp1251_cs_as = LOWER(p.nome) collate
sql_latin1_general_cp1251_cs_as
WHERE dboc.natureza =
'MONOGRAFIA_DE_CONCLUSAO_DE_CURSO_APERFEICOAMENTO_E_ESPECIALIZACAO'
UNION ALL
SELECT dboc.titulo, dboc.ano, detoc.nome_do_orientado, detoc.nome_da_instituicao,
detoc.nome_do_curso, 'Graduacao' AS tipo, dg.nome_completo,
cv.numero_identificador, p.sigla_centro
FROM lattes.outras_orientacoes_concluidas oc
JOIN lattes.dados_basicos_de_outras_orientacoes_concluidas dboc ON
dboc.id_dados_basicos_de_outras_orientacoes_concluidas =
oc.id_dados_basicos_de_outras_orientacoes_concluidas
JOIN lattes.detalhamento_de_outras_orientacoes_concluidas detoc ON
detoc.id_detalhamento_de_outras_orientacoes_concluidas =
oc.id_detalhamento_de_outras_orientacoes_concluidas
JOIN lattes.orientacoes_concluidas ocs ON ocs.id_orientacoes_concluidas =
oc.id_orientacoes_concluidas
JOIN lattes.outra_producao op ON op.id_outra_producao = ocs.id_outra_producao
JOIN lattes.curriculo_vitae cv ON cv.id_curriculo_vitae = op.id_curriculo_vitae
JOIN lattes.dados_gerais dg ON dg.id_curriculo_vitae = cv.id_curriculo_vitae
LEFT JOIN lattes.professores p ON LOWER(dg.nome_completo) collate
sql_latin1_general_cp1251_cs_as = LOWER(p.nome) collate
sql_latin1_general_cp1251_cs_as
WHERE dboc.natureza = 'TRABALHO_DE_CONCLUSAO_DE_CURSO_GRADUACAO'
UNION ALL
SELECT dboc.titulo, dboc.ano, detoc.nome_do_orientado, detoc.nome_da_instituicao,
detoc.nome_do_curso, 'Iniciacao cientifica' AS tipo, dg.nome_completo,
cv.numero_identificador, p.sigla_centro
FROM lattes.outras_orientacoes_concluidas oc
JOIN lattes.dados_basicos_de_outras_orientacoes_concluidas dboc ON
dboc.id_dados_basicos_de_outras_orientacoes_concluidas =
oc.id_dados_basicos_de_outras_orientacoes_concluidas
```

```
JOIN lattes.detalhamento_de_outras_orientacoes_concluidas detoc ON
detoc.id_detalhamento_de_outras_orientacoes_concluidas =
oc.id_detalhamento_de_outras_orientacoes_concluidas
JOIN lattes.orientacoes_concluidas ocs ON ocs.id_orientacoes_concluidas =
oc.id_orientacoes_concluidas
JOIN lattes.outra_producao op ON op.id_outra_producao = ocs.id_outra_producao
JOIN lattes.curriculo_vitae cv ON cv.id_curriculo_vitae = op.id_curriculo_vitae
JOIN lattes.dados_gerais dg ON dg.id_curriculo_vitae = cv.id_curriculo_vitae
LEFT JOIN lattes.professores p ON LOWER(dg.nome_completo) collate
sql_latin1_general_cp1251_cs_as = LOWER(p.nome) collate
sql_latin1_general_cp1251_cs_as
WHERE dboc.natureza = 'INICIACAO_CIENTIFICA'
UNION ALL
SELECT dboc.titulo, dboc.ano, detoc.nome_do_orientado, detoc.nome_da_instituicao,
detoc.nome_do_curso, 'Outro' AS tipo, dg.nome_completo, cv.numero_identificador,
p.sigla_centro
FROM lattes.outras_orientacoes_concluidas oc
JOIN lattes.dados_basicos_de_outras_orientacoes_concluidas dboc ON
dboc.id_dados_basicos_de_outras_orientacoes_concluidas =
oc.id_dados_basicos_de_outras_orientacoes_concluidas
JOIN lattes.detalhamento_de_outras_orientacoes_concluidas detoc ON
detoc.id_detalhamento_de_outras_orientacoes_concluidas =
oc.id_detalhamento_de_outras_orientacoes_concluidas
JOIN lattes.orientacoes_concluidas ocs ON ocs.id_orientacoes_concluidas =
oc.id_orientacoes_concluidas
JOIN lattes.outra_producao op ON op.id_outra_producao = ocs.id_outra_producao
JOIN lattes.curriculo_vitae cv ON cv.id_curriculo_vitae = op.id_curriculo_vitae
JOIN lattes.dados_gerais dg ON dg.id_curriculo_vitae = cv.id_curriculo_vitae
LEFT JOIN lattes.professores p ON LOWER(dg.nome_completo) collate
sql_latin1_general_cp1251_cs_as = LOWER(p.nome) collate
sql_latin1_general_cp1251_cs_as
WHERE dboc.natureza = 'ORIENTACAO-DE-OUTRA-NATUREZA') x
```

Anexo III

Artigo

Ultrapassando o poder de expressão de SQL com mineração de dados

Katiany Zimmermann[1], Carina F. Dorneles[1]

[1]Departamento de Informática e Estatística – Universidade Federal de Santa Catarina (UFSC)
Florianópolis – SC – Brasil

kati.zim@grad.ufsc.br,dorneles@inf.ufsc.br

Abstract. *Although SQL is a powerful language with great expression, through its algebraic operators and relational calculus, some limitations occur while trying to find data patterns. Based on this, this papers aims to demonstrate how far SQL can be used exclusively to extract data information from a database. Some experiments will be executed in order to express what can and what cannot be done with SQL. When finding questions that cannot be answered with SQL, data mining technics will be applied.*

Resumo. *Apesar do forte poder de expressão do SQL, que se manifesta através de operadores algébricos e cálculos relacionais, é possível observar limitações quando o objetivo se dá em deduzir padrões sobre os dados. Com base nisto, este trabalho visa explorar, através de experimentos, até onde é possível obter as informações desejadas sobre um banco de dados fazendo uso exclusivamente de SQL. A partir das limitações encontradas, serão descritas soluções com mineração de dados, de forma a evidenciar motivações para seu uso.*

1. Introdução

Bancos de dados são utilizados em diferentes domínios e aplicações para organizar e armazenar os dados [3]. O uso de banco de dados permite, através de sua organização, controlar a integridade, o acesso e as transações realizadas sobre os dados [11]. A procura de dados através de consultas, por sua vez, se torna mais eficiente, permitindo programas de computadores conseguem buscar dados de forma rápida e simplificada [9]. A representação de tais dados se dá através de modelos. O modelo relacional, por exemplo, é altamente utilizado, uma vez que possui uma definição concreta e madura, oferecendo controle sobre os dados [3].

Além do uso de programas, com diferentes linguagens, a busca de dados em um banco de dados relacional pode ser feita de linguagens de consulta estruturadas (SQL). Sendo um padrão ANSI[1] desde 1986 estruturado pela ISO[2] desde 1987, SQL é

---

[1] *American National Standard Institute*. Em português: Instituto Nacional Americano de Padrões.

[2] *International Organization for Standardization*. Em português: Organização Internacional para Padronização.

considerado um padrão mundialmente. Além disso, não é necessário programar grandes aplicações para usar SQL, que resolve consultas por si de maneira rápida e declarativa para consultar os dados.

Em bancos de dados com características de dados faltantes, ruídos e dados escondidos, apesar de sua excelência, SQL não tem poder para extrair análises estatísticas faltantes ou descobrir padrões, como grupos ou informações similares. Um exemplo real deste cenário é o sistema IPU[3], uma plataforma para consulta a dados cadastrados por membros da UFSC no currículo Lattes [10]. Com os dados do IPU, são gerados gráficos indicativos de produção intelectual dos pesquisadores da UFSC [10]. Nos currículos dos pesquisadores, a informação de departamento, por exemplo, não está explícita. Desta forma, não é possível agrupar os professores por departamento fazendo uso apenas de SQL.

O uso de algoritmos de mineração de dados permite extrair e interpretar padrões dos dados [1][4][7]. A mineração de dados permite, através de algoritmos automatizados, extrair informações úteis sobre grandes bancos de dados [2][6]. É preciso, no entanto, conhecer o contexto ao qual se deseja aplicar os algoritmos e entender os dados trabalhados.

Considerando as abordagens até então descritas, este artigo visa realizar experimentos explorando o poder de expressão de SQL, explorando aplicações com técnicas de mineração de dados em situações não resolvidas por SQL. O sistema IPU é utilizado como fonte de dados, onde buscas de dados serão realizadas através das duas abordagens. Este trabalho tem como contribuição para o estado da arte um estudo referente ao poder de expressão de SQL através de experimentos sobre a base de dados do IPU. Em virtude disso, para buscar respostas em situações não suportadas por SQL, serão aplicadas técnicas de mineração de dados.

Este artigo está organizado em 4 seções. A primeira contém a introdução até então apresentada. A segunda seção apresenta técnicas que expressam o poder de SQL. Na terceira seção, buscas que não podem ser executadas apenas com SQL. A quarta seção mostra o uso de algoritmos de mineração de dados. Por fim, a quinta seção descreve as conclusões e trabalhos futuros.

2. Técnicas que expressam o poder de SQL

Nesta seção, são exploradas duas técnicas a serem realizadas sobre os dados com o uso de SQL: agregação ou agrupamento de valores e similaridades.

2.1 Agregação (ou agrupamento) de valores

Para explorar tal técnica, busca-se gerar um relatório, para cada ano dos últimos 5 anos, com os 3 centros que tiveram o maior número de produção de patentes, incluindo o número de patentes produzidas.

---

[3] http://ipu.sistemas.ufsc.br

Possível desafio em SQL: organizar os dados de forma a extrair apenas os três centros com maior número de produção de patentes na resposta da consulta.

Query SQL:

```
SELECT ano, centro, patentes FROM (
SELECT patentes, centro, ano, row_number() OVER
      (PARTITION BY ano ORDER BY patentes DESC) AS rnk
      FROM (
      SELECT count(*) AS patentes, sigla_centro as centro, ano
            FROM lattes.vw_patentes
            GROUP BY sigla_centro
) AS pat
      WHERE ano BETWEEEN 2011 AND 2015 AND centro IS NOT NULL) AS res
WHERE rnk <=3
ORDER BY ano DESC, patentes DESC
```

Solução: É possível, utilizando a visão `patentes`, contar o número de patentes de um ano para cada centro, utilizando as funções `COUNT`, sobre o título da patente e `GROUP BY`, sobre o ano. Para filtrar apenas os anos de 2011 a 2015, utiliza-se a cláusula `WHERE` condicionando que o valor de ano esteja entre 2011 e 2015. Para selecionar apenas os 3 centros que mais produziram patentes, a função `ROW_NUMBER()` é utilizada, associando um número, iniciado em 1, em ordem crescente, para cada resultado dentro dos diferentes anos, ordenado pela quantidade de produções. Esse número é inserido na cláusula `WHERE` para limitar os resultados até a terceira linha de cada ano. Desta forma, conclui-se que SQL é capaz de extrair os dados do enunciado, conforme resultado da Figura 1.

	ano	centro	patentes
1	2015	CTC	50
2	2015	CCE	18
3	2015	CCS	8
4	2014	CTC	50
5	2014	CFM	10
6	2014	JOI	8
7	2013	CTC	49
8	2013	JOI	11
9	2013	CFM	11
10	2012	CTC	50
11	2012	CFM	10
12	2012	CCS	7
13	2011	CTC	47
14	2011	CSE	26
15	2011	CCB	12

Figura 1. Relatório dos três centros com mais patentes nos últimos cinco anos.

2.1 Similaridade entre dados que se referem ao mesmo objeto com diferentes representações

Busca-se gerar um relatório do número de orientações concluídas do curso de Sistemas de Informação.

Possível desafio em SQL: considerar todos os registros possíveis do nome do curso, incluindo variações e erros de digitação.

Query SQL:

```
SELECT nome_do_curso, count(nome_do_curso) ocorrências
FROM dbufsc.lattes.vw_orientacoesConcluidas
WHERE nome_do_curso like '%sist%informac%'
GROUP BY nome_do_curso
UNION
SELECT n, c
FROM (
      SELECT edit_distance_within('%sistemas de informacao', n,10)
      d, n, c
      FROM (
      SELECT count(nome_do_curso) c, nome_do_curso n
      FROM [dbufsc].[lattes].vw_orientacoesConcluidas
      GROUP BY nome_do_curso
      ) grouped
) similar WHERE d>=0
ORDER BY ocorrencias DESC
```

Solução: utilizando a função que implementa o algoritmo da distância de Levenshtein [5][8]. São considerados os registros de cursos similares à *string* "Sistemas de Informação". Com tal função, é possível obter um número de 1453 orientações concluídas para o curso especificado.

3. Buscas não efetuadas com SQL

Os enunciados construídos neste tópico (EDC) demonstram as situações de buscas que vão além do poder de expressão de SQL, explorando os ruídos dos dados presentes no IPU.

EDC1: Gerar um relatório de todos os professores por departamento.

Origem do problema: Nos currículos Lattes, não há um campo de preenchimento específico para departamento. Uma análise dos dados permitiu concluir que alguns professores preenchem tal informação em diferentes partes do currículo, como no resumo, no endereço profissional e em vínculos. Com uma consulta em SQL, é possível obter a informação de departamento de 2304 currículos, de um total de 4738 currículos existente na base de dados. Ou seja, apenas 48% dos professores registram o seu departamento no currículo, em locais diferentes.

Limitação encontrada: Apesar de haver informações de departamento no banco de dados do IPU, não é possível, com SQL, extrair o departamento de 52% dos professores.

EDC2: Predizer o número de orientações concluídas ao final de 2016.

Origem do problema: Uma vez que 2016 é o ano atual, orientações que estão por ser concluídas constam ainda como "em andamento" no IPU. No entanto, a falta de atualizações dos currículos faz com que o número de orientações, tanto em andamento como concluídas, exibido no IPU, não sejam suficientes para saber o número de orientações concluídas ao final de 2016. Para ter uma estimativa do número de orientações que serão concluídas ao final do ano, é preciso fazer predições. No IPU, as orientações em andamento

Limitação encontrada: Assim como descrito na limitação do problema anterior (EDC3), não é possível, tecnicamente, com SQL, predizer valores de dados e informações que não existem no banco de dados.

4. Aplicação de Técnicas de Mineração de Dados

A seguir, são propostas soluções para os enunciados do tópico anterior, utilizando algoritmos de mineração de dados com a ferramenta RapidMiner. Serão aplicadas as técnicas de classificação e predição, descritas na seção 2.1.2 deste trabalho.

4.1. Classificação dos professores por departamento

Foi aplicado o classificador bayesiano para gerar o modelo, com 51 segundos de execução.

Modelo gerado: o modelo gerado resultou em pelo menos 139 classes com probabilidades > 0, sendo 73 classes com probabilidade > 0.002 e 17 mais relevantes com probabilidade > 0.01.

Observações do resultado: antes de aplicar o algoritmo, o maior número de professores com valor de departamento registrado no banco se concentrava no departamento de química. Com a aplicação do algoritmo mineração de dados, por sua vez, os grupos de departamentos ficaram mais distribuídos.

4.2 Predição do número de orientações concluídas em 2016

Para este problema, foi aplicado o método de regressão linear.

Os dados foram divididos em conjunto de treinamento (95% dos dados, com exemplares lineares para realizar o aprendizado ao decorrer dos anos) e conjunto de teste; aplicação do algoritmo de regressão linear para gerar modelo, com 5 segundos de execução.

Observações do resultado: como mostra a Figura , o algoritmo conseguiu predizer um valor de 9.834 orientações concluídas ao final de 2016. No entanto, observou-se, no enunciado anterior, que o número de orientações concluídas de 2015 pode ser maior do que o presente no banco de dados, devido a falta de atualizações dos currículos Lattes. Desta forma, vale adaptar as configurações da etapa de mineração para considerar um conjunto de treinamento menor, aumentando o conjunto de teste para aplicar as previsões desde 2015. Neste caso, com um conjunto de treinamento correspondente a 90% dos registros, a Figura 2 resulta em um número consideravelmente maior de

17.792 orientações concluídas em 2016. Assim como no enunciado anterior, o número de professores admitidos e desligados da UFSC poderiam auxiliar o aprendizado do algoritmo.

ano	andamento	concluidas	prediction(c...
2011	960	15050	?
2012	2207	14462	?
2013	3109	14989	?
2014	5319	12965	?
2015	7173	9413	?
2016	1491	1011	9834.619

Figura 2. Predição das orientações concluídas em 2016.

5. Conclusões e Trabalhos Futuros

A transformação dos dados em informação e conhecimento deve ser prioridade em um ambiente de banco de dados. Grandes volumes de dados requerem ambientes bem estruturados, o que inclui a organização dos dados e o conhecimento das ferramentas disponíveis. Um ambiente de desenvolvimento pode iniciar basicamente com um banco de dados e um sistema de acesso à informação. Com o aumento da quantidade de dados, pode ser conveniente ampliar tal ambiente, aderindo novas ferramentas e explorando diferentes conceitos, como a mineração de dados.

Em um sistema de gerenciamento de banco de dados, SQL conta alto poder de expressão para extrair dados. Incontáveis consultas podem ser realizadas sobre um banco de dados utilizando declarações com operadores e funções de SQL. No entanto, apesar de uma estrutura bem definida de um banco de dados, dados inconsistentes podem começar a surgir de modo que SQL deixe de ser uma solução.

O uso de outras abordagens pode ultrapassar o poder de expressão de SQL. A mineração de dados, através de diferentes algoritmos, é capaz de aprender os padrões implícitos nos dados e realizar descoberta de conhecimento. Estudos relacionados mostram que o uso de SQL ocorre até o momento do preparo dos dados, para, a partir de então, a aplicação de novos conceitos. Os dados obtidos através de consultas SQL, em uma etapa de pré-processamento, são utilizados em ferramentas de mineração. Sendo assim, as limitações presente em SQL são supridas com um novo conceito em novas etapas.

Através dos experimentos realizados explorando SQL e mineração de dados, este trabalho corrobora com o estado da arte. Apesar das dificuldades encontradas em SQL, as possibilidades de consultas e operações sobre os dados são inúmeras. Com grandes quantidades de dados, SQL e mineração de dados são duas abordagens diferentes, capazes de extrair dados de diferentes maneiras.

Como trabalhos futuros, novas pesquisas, experimentos e aplicações podem ser realizadas. No IPU, por exemplo, podem ser incluídas ferramentas de mineração, de modo a gerar novos relatórios com análises qualitativas e quantitativas, auxiliando na

tomada de decisão. Outros dados podem ser vinculados ao IPU, aperfeiçoando o aprendizado dos algoritmos. Diferentes técnicas que exploram poder de processamento e memória computacionais podem ser aplicadas para obter melhores resultados.

Referências

[1] BIGOLIN, Nara Martini; BOGORNY, Vania; ALVARES, Luis Otávio. Uma Linguagem de Consulta para Mineração de Dados em Banco de Dados Geográficos Orientado a Objetos. In: Conferencia Latinoamericana de Informatica. 2003. p. 23-35.

[2] BOGORNY, Vania. Algoritmos e ferramentas de descoberta de conhecimento em bancos de dados geográficos. 2003. Dissertação - PPGC, UFRGS, Porto Alegre, 2003.

[3] ELMASRI, Ramez. Fundamentals of database systems. Pearson Education India, 2010.

[4] FAYYAD, Usama; PIATETSKY-SHAPIRO, Gregory; SMYTH, Padhraic. From data mining to knowledge discovery in databases. AI magazine, v. 17, n. 3, p. 37, 1996.

[5] FRIBBLE, Arnold. SQL Server Forums – Levenshtein Edit Distance Algorithm, 2005. Disponível em: <http://www.sqlteam.com/forums/topic.asp?TOPIC_ID=51540>. Acesso em: 15 de ago. 2016.

[6] GORUNESCU, Florin. Data Mining: Concepts, models and techniques. Springer Science & Business Media, 2011.

[7] HAN, Jiawei; KAMBER, Micheline. Data mining concepts and techniques. Simon Fraser University, 2000.

[8] LEVENSHTEIN, Vladimir I. Binary codes capable of correcting deletions, insertions, and reversals. Soviet Physics Doklady, v. 10, p. 707, 1966.

[9] RAMAKRISHNAN, Raghu; GEHRKE, Johannes. Database management systems. McGraw-Hill, 2003.

[10] SEGALIN, Vinicius S. Ipu: sistema web de acesso a dados de Currículos Lattes a partir do mapeamento XML-Relacional dos currículos de professores e pesquisadores da UFSC. 2014. Dissertação (Graduação em Sistemas de Informação) - UFSC, Florianópolis, 2014.

[11] SUMATHI, Sai; ESAKKIRAJAN, S. Fundamentals of relational database management systems. Springer, 2007.